NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

DISSERTATION

CHANCE-CONSTRAINED MISSILE-PROCUREMENT AND DEPLOYMENT MODELS FOR NAVAL SURFACE WARFARE

by

Ittai Avital

March 2005

Dissertation Supervisors: R. Kevin Wood
Moshe Kress

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2005	**3. REPORT TYPE AND DATES COVERED** Dissertation

4. TITLE AND SUBTITLE: Chance-Constrained Missile-Procurement and Deployment Models for Naval Surface Warfare	**5. FUNDING NUMBERS**
6. AUTHOR Ittai Avital	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	**12b. DISTRIBUTION CODE**

13. ABSTRACT

We model the problem of minimum-cost procurement and allocation of anti-ship cruise missiles to naval combat ships as a two-period chance-constrained program with recourse. Discrete scenarios in two periods define “demands” for missiles (i.e., targets and number of missiles required to kill those targets), which must be met with acceptable probabilities. After the first combat period, ships may replenish their inventories from a depot, if the depot’s inventory suffices. A force commander assigns targets to ships based on missile load-outs and target demands.

The deterministic-equivalent integer program solves too slowly for practical use. We propose a specialized decomposition algorithm, implemented in MATLAB, which solves the two-period model via a series of single-period problems. The algorithm yields optimal solutions for a wide range of missile-allocation directives, and usually near-optimal solutions otherwise. We exploit the fact that each single-period problem is a probabilistic integer program, whose solution must be a p-efficient point (PEP) of that period’s demand distribution. Our algorithm uses PEP-enumeration techniques developed by Beraldi and Ruszczyński, and a specialized algorithm from Kress, Penn and Polukarov. The algorithm solves real-world problem instances in a few minutes or less.

14. SUBJECT TERMS Inventory Models, Target Assignment, Stochastic Programming, Probabilistic Programming			**15. NUMBER OF PAGES** 147
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

CHANCE-CONSTRAINED MISSILE-PROCUREMENT AND DEPLOYMENT MODELS FOR NAVAL SURFACE WARFARE

Ittai Avital
B.S., Hebrew University of Jerusalem, 1995
M.Sc., Tel Aviv University, 2002
M.Sc., National University of Singapore, 2004
M.Sc., Naval Postgraduate School, 2004

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 2005

Author: ____________________
Ittai Avital

Approved by:

R. Kevin Wood Professor of Operations Research Dissertation Supervisor Committee Chairman	Moshe Kress Professor of Operations Research Dissertation Supervisor
Guillermo Owen Distinguished. Professor of Mathematics	Javier Salmeron Research Assistant Professor of Operations Research
Wayne P. Hughes Dean, Graduate School of Operational and Information Sciences	

Approved by: ____________________
James Eagle, Chairman, Department of Operations Research

Approved by: ____________________
Julie Filizetti, Associate Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

We model the problem of minimum-cost procurement and allocation of anti-ship cruise missiles to naval combat ships as a two-period chance-constrained program with recourse. Discrete scenarios in two periods define “demands” for missiles (i.e., targets and number of missiles required to kill those targets), which must be met with acceptable probabilities. After the first combat period, ships may replenish their inventories from a depot, if the depot’s inventory suffices. A force commander assigns targets to ships based on missile load-outs and target demands.

The deterministic-equivalent integer program solves too slowly for practical use. We propose a specialized decomposition algorithm, implemented in MATLAB, which solves the two-period model via a series of single-period problems. The algorithm yields optimal solutions for a wide range of missile-allocation directives, and usually near-optimal solutions otherwise. We exploit the fact that each single-period problem is a probabilistic integer program, whose solution must be a p-efficient point (PEP) of that period’s demand distribution. Our algorithm uses PEP-enumeration techniques developed by Beraldi and Ruszczyński, and a specialized algorithm from Kress, Penn and Polukarov. The algorithm solves real-world problem instances in a few minutes or less.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ASCM	Anti-Ship Cruise Missile
CCGIM	Chance-Constrained Ground Inventory Model
CCIM	Chance-Constrained Inventory Model
CCNIM	Chance-Constrained Naval Inventory Model
CCNIM-dc	**CCNIM** decomposition
CCNIM-MAP	**CCNIM** with MAP
CCNIM-sp	**CCNIM** single period
CCNIM-pII	**CCNIM** in period-II
CCNIM-pIIm	**CCNIM** in period-II modified
DPLP	Discrete Probabilistic Linear Program
FFAM	Fully-Flexible Assignment Model
FMSP	Flexible Minmax Subset Problem
FMSP-lb	**FMSP** lower bound
IP	Integer Program
KPP	Kress, Penn and Polukarov
LP	Linear Programming
MAP	Monotonic Assignment Policy
MSP	Minmax Subset Problem
MSPA	Minmax Subset Problem Algorithm
PEP	P-Efficient Point
PIP	Probabilistic Integer Program
RFFAM	Reduced Fully-Flexible Assignment Model
RFFAM-lb	**FFAM** lower bound
RFFAM-mII	**RFFAM** monotonic (allocation in period) II
RFFAM-rx	**RFFAM** relaxation
RFFAM-sp	**RFFAM** single period
RFFAM-spII	**RFFAM** single period (for period) II

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

This work was born of most unlikely circumstances, as I was originally scheduled for a one-year stay in Monterey to obtain a Master's degree in Operations Research. I would like to thank Professor Tom Lucas for first suggesting that I stay for a PhD, and then outlining the scheme by which it could be done. I would also like to thank Chairman Jim Eagle for his great efforts in pushing my unusual request through the bureaucratic maze. Without these professors' help, I would never have been in a position to pursue this research.

Of course, I would have never been able to complete this work without the aid and guidance of my advisors, Professors Kevin Wood and Moshe Kress, who spent many hours reading, polishing, pointing out errors, and correcting my rough ideas. This dissertation is so much better for their effort.

I would also like to thank Professors Guillermo Owen and Javier Salmeron, and Dean Wayne Hughes, for their insightful comments regarding this work, and Professors Robert Read, Bob Koyak, and Matt Carlyle for going out of their way to educate me in the field of OR.

I dedicate this work to Raadthie, my love, who sweetened my stay so far from home, and to my parents, who have educated me in so many other things.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

We present models for the procurement and allocation of anti-ship cruise missiles, a difficult problem facing many modern navies, which must plan for many possible combat scenarios. The number of missiles required by each ship over an entire conflict may exceed its capacity, so a ship may need to return to a depot at a port to load more missiles between successive periods of combat. We consider two periods of combat. "period I" and "period II," and seek to determine minimum-cost initial ship load-outs plus depot-level inventory, while ensuring, with sufficiently high probability, that all ships can satisfy their assigned missions for each combat period. Each mission represents a target and its associated "demand," which is the number of missiles required to successfully engage that target.

We formulate this problem as a two-period, three-stage, chance-constrained inventory model, which we denote by **CCNIM** (Chance-Constrained Naval Inventory Model). We assume that all targets will be in range of each combat ship in the fleet, and the force commander can assign any target to any ship; however, the force commander assigns at most one target to each ship, and does so based on the available missile inventories and the target demands. We further assume that ships have bounded capacities, carry no safety stocks, are not recalled to port to offload excess inventories prior to period II, and there is no direct transshipment of missiles between ships. A "cost ratio," the cost of a missile stored in the depot for potential use in period II divided by those initially allocated to the ships, reflects operational preferences rather than actual monetary costs.

To solve **CCNIM**, we first formulate it as **FFAM** (Fully-Flexible Assignment Model), a mixed-integer program with enumerated scenarios**.** Because we expect the force commander to follow some assignment heuristic when assigning targets to ships, we add constraints to **FFAM** that assign targets with larger demands to ships with larger inventories. We refer to this assignment policy as the "MAP," and show that assigning targets according to this policy satisfies any single-period scenario that can possibly be satisfied with the existing ship inventory levels. Consequently, it is the best myopic

policy a force commander can adopt. We also prove that the inventories will be sufficient to sustain the two periods of combat for an arbitrary assignment plan if transshipment is allowed.

Clearly, robustness can always be achieved by procuring large quantities of missiles. However, if targets are assigned according to the MAP in every scenario, then the number of missiles we must allocate to cover the random demands with the required probability of success is minimal in any particular combat period. Empirically, we observe that enforcing the MAP rarely increases the cost of an optimal solution in **FFAM**. We adopt **CCNIM-MAP,** a version of **CCNIM** that includes the MAP, as our baseline model.

Although enforcing the MAP in **FFAM** improves solution times significantly, **FFAM** remains too difficult to solve for cases of practical size. For example, an instance of **FFAM,** involving six ships and five scenarios in each period, generates 10,305 equations and 14,672 variables in the integer programming model. It can be solved in less than an hour only when the cost ratio is very large or very small. We therefore require better solution methods to solve **CCNIM-MAP**.

We propose **CCNIM-dc**, a decomposition algorithm to solve **CCNIM-MAP.** The solution is provably optimal for every cost ratio not greater than 1, and if the cost ratio is sufficiently greater than 1. In other cases, the solution is not provably optimal, but **CCNIM-dc** also provides a lower bound on the optimal cost, so an optimality gap can be calculated. In most of the cases examined, the relative optimality gap is only a few percent. However, the optimality gap cannot be reduced by further computation. If the gap is deemed too large, less efficient techniques must be used to solve that instance.

We exploit the fact that the problem of allocating sufficient missiles to satisfy a single period of combat, when the MAP is enforced, is an instance of a probabilistic integer program (PIP), as described by Beraldi and Ruszczyński. The feasible region of a PIP has a special structure that is defined by a set of “p-efficient points” (PEPs), a concept developed by Dentcheva, Prékopa and Ruszczyński. **CCNIM-dc** enumerates a relatively small, cost-ratio-dependent set of period-I allocations, and calculates the minimum depot inventory that follows from each one. Minimum depot inventories are found either by enumerating PEPs, or by an algorithm proposed by Kress et al. for a ground-combat version of **CCNIM**. **CCNIM-dc** is not based on linear or integer programming, and can be implemented using a standard programming language.

We implement **CCNIM-dc** in MATLAB^{TM} version 7.0 on a personal computer with a 2.8 GHz Intel Pentium IV processor. Its solution times are faster than those of **FFAM** by several orders of magnitude. **CCNIM-dc** solves in a few milliseconds instances that **FFAM** cannot solve in an hour, and can solve instances of practical size in a reasonable length of time. For example, cases with as many as 25 ships and 25 scenarios in each period usually solve in less than 100 seconds, and never require more than 1,000 seconds.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Two of the key questions that military logisticians ask are: How much ordnance should be procured, and how should it be distributed to combat units given the uncertainty of combat? This dissertation develops models and solution methods to help answer instances of these questions in the context of naval surface warfare.

A. A GENERAL SUPPLY-CHAIN FRAMEWORK FOR COMBAT UNITS

Combat forces routinely use a variety of ordnance to suppress enemy forces, destroy them, or disrupt their weapons. Sustaining a combat force requires plans to prescribe the total amount of ordnance of each type that should be procured, the initial allocation plan to combat units, and, by implication, the amount of ordnance to be stored in one or more depots for distribution at a later time. An optimal plan must identify a minimum-cost package of ordnance that meets operational requirements with a sufficiently high probability of success. Uncertainty arises because a variety of combat scenarios may arise, each with potentially distinct demands for ordnance.

We define a combat scenario, or simply *scenario*, as a set of demands for ordnance in a single period of combat. A *demand* denotes the amount of ordnance that must be fired by a *shooter* (friendly unit) to successfully prosecute its assigned mission in that scenario. A *mission* typically consists of one or more *targets* (enemy units) that must be prosecuted. Because the amount of ordnance fired in a successful engagement of a specific target is essentially random [Ancker 1982, pp. 1-8], we may represent a single envisioned set of targets by several different scenarios. Note that we only consider demands for a single type of ordnance, such as artillery shells or anti-ship cruise missiles (ASCMs). The duration of a combat period is not specified in days, but rather is the time between successive opportunities to replenish supplies. In the naval context, for example, a combat period can last several days, and is delimited by “lulls” in the combat, during which some or all of the ships in a fleet could return to port for resupply.

For a force to *successfully* prosecute a scenario, we assume that all demands must be met using the units’ available supply of ordnance. We refer to successfully prosecuted scenarios as *satisfied* scenarios. If any unit cannot satisfy the demand for its assigned mission, the scenario is considered *unsatisfied*. It is possible to relax this assumption and require that only a fixed fraction of the demands be met to satisfy a scenario. We do not consider this model variant in this dissertation, however.

The planning horizon may include several periods of combat in succession, and the probability that a particular scenario in period $t > 1$ occurs can be conditional on the scenario that actually occurred in period t–1. We refer to the sequence of single-period scenarios for $t = 1,...,T$ as a *compound scenario*. Note that in most stochastic-programming literature, what we call a "scenario" is referred to as a "node in a scenario tree," and our "compound scenario" is simply a "scenario." Our terminology is more convenient when considering multi-period problems as a series of single-period problems, a recurring idea in this dissertation.

We assume that the scenarios, with their respective probabilities, have been established by planners as an extension of the war plans. The scenarios can represent different hypothesized battles as well as uncertain demands within each battle. The duration of the conflict we plan for is usually long enough that combatants have the opportunity to replenish their supplies, often more than once [Rabinovitch 1997, p. 252].

Each combat unit has some inventory capacity associated with it. The capacity may be "strict," as is the case for missiles carried aboard combat ships, or "elastic," as is the case for many ground units, which can increase the number of supply vehicles traveling with the combatants. A unit's total requirement for ordnance to satisfy its forecasted assigned missions in a compound scenario may exceed its carrying capacity. In that case, the unit may be forced to replenish its supply after some period of combat.

Even if a unit's carrying capacity is sufficient to meet the largest conceivable compound demand, it may be preferable not to load the unit to that level. Following the realization of a scenario in some period, the probabilities of scenarios in subsequent period are updated. It is possible then that high-demand scenarios no longer seem likely, and the requirements for certain units can be lowered. We wish to avoid situations in which, after a period of combat, some units hold excess ordnance that may be needed by other units in the next period. Replenishment from central locations is advantageous then, since it utilizes *risk pooling* (e.g., Simchi-Levi *et al.* [2000, pp. 56-60]) to reduce the total amount of ordnance that must be deployed in the first combat period. Note that risk pooling refers to the risk of shortages due to high demands, and not to the risk of losing ordnance associated with disabled units. (It is clear that reducing the amount of deployed ordnance reduces the amount of ordnance that can be lost to enemy actions, but we do not address such issues in this dissertation.)

One of the common features of our problem and stochastic supply-chain models in the civilian sector is *recourse*. After a period of combat, we can resupply ships in order to improve our chances for success in subsequent periods. This action constitutes recourse, as defined in the stochastic-programming literature (e.g., Birge and Louveaux [1997, pp. 84-100, 122-127]). Thus, the model we eventually create will be a multi-stage stochastic-programming model with recourse.

B. CHANCE-CONSTRAINED PROGRAMMING

One of the key features that distinguishes the problem of procuring and deploying ordnance from other supply problems is the singularity of war [Kress 2002, p. 242]. This contrasts with civilian supply-chain planning models that typically apply to repetitive scenarios over relatively long time horizons [Diwekar 2002]. Because we expect the planned supply chain to be tested in war only once, albeit across multiple time periods, our models will incorporate probability requirements, i.e., *probabilistic constraints*, or *chance constraints*. (For example, see Birge and Louveaux [1997, pp. 103-108].) Multi-stage supply-chain models using chance constraints to guarantee service levels do exist (e.g., Charnes, Cooper and Symonds [1958], Murr and Prékopa [1996], Bassok et al. [2002], Paschalidis et al. [2004]), but a typical multi-stage stochastic program would evaluate the effect of uncertainty through an objective function involving expected costs and/or penalties (see Porteus [1990] and the references therein) as opposed to using chance constraints.

The first mathematical-programming model to incorporate chance constraints is attributed to Charnes et al. [1958], who model the multi-period planning of heating-oil production under uncertain demand. They introduce multiple individual chance constraints to guarantee each period's demand is met with the required probability. Miller and Wagner [1965] introduce joint chance constraints as a theoretical extension of the work of Charnes et al. Since the 1960s, the theoretical understanding of chance constraints has increased significantly (see Prékopa [1995] and the references therein), and several specialized algorithms for solving problems formulated with such constraints have been proposed. However, most of this research has been restricted to models involving continuous distributions and decision variables

When the probability distributions are discrete, or the decision variables integer, chance-constrained models become significantly harder to solve [Beraldi and Ruszczyński 2001], and there is relatively little literature dedicated to such models. Dentcheva, Prékopa and Ruszczyński [2000] provide an extensive analysis of the properties of the feasible region of the discrete probabilistic linear program (DPLP) with a joint chance constraint involving the discrete random vector $\boldsymbol{\xi}$:

(DPLP)

$$\min \ \mathbf{c}^T\mathbf{x} \tag{1.1}$$

s.t.

$$\Pr\left\{T\mathbf{x} \geq \boldsymbol{\xi}\right\} \geq p \tag{1.2}$$

$$A\mathbf{x} \geq \mathbf{b} \tag{1.3}$$

$$\mathbf{x} \geq \mathbf{0} \tag{1.4}$$

To analyze this problem, the authors use the concept of a p-efficient point, first introduced by Prékopa [1990]:

Definition: Let $p \in [0,1]$, and let F denote the probability distribution for the *n*-dimensional, discrete random vector $\boldsymbol{\xi}$. A point $\mathbf{v} \in \mathbb{R}^n$ is called a *p-efficient point* (PEP) of F, if $F(\mathbf{v}) \geq p$ and there is no $\mathbf{y} \leq \mathbf{v}$, $\mathbf{y} \neq \mathbf{v}$ such that $F(\mathbf{y}) \geq p$. ■

Dentcheva et al. prove that there is a positive and finite number of PEPs associated with any discrete random vector, and that the feasible region for the probabilistic constraint in any DPLP can be reformulated as a union of cones emanating from those PEPs. In particular, let $\mathbf{v}_j$, $j \in J$, be all the PEPs of F and let $K_j = \mathbf{v}_j + \mathbb{R}^n_+$ denote the cone emanating from $\mathbf{v}_j$; then constraint (1.2) is equivalent to

$$T\mathbf{x} \in \bigcup_{j \in J} K_j \ . \tag{1.5}$$

We can solve a DPLP by enumerating all its PEPs, and by then processing the simple linear programs that result from considering each cone separately. Prékopa, Vizvári and Badics [1996] and Beraldi and Ruszczyński [2001] develop algorithms for enumerating PEPs; Beraldi and Ruszczyński [2002] provide additional details. If ξ has many dimensions, the algorithms may be computationally expensive, so the authors propose iterative methods that generate new PEPs as needed, en route to finding an optimal solution.

Replacing constraints (1.4) with the integrality restriction $\mathbf{x} \in \mathbb{Z}_+^n$ creates a variation of a DPLP that Beraldi and Ruszczyński [2001] call a *probabilistic integer program* (PIP). These authors use the PEPs of F in a specialized branch-and-bound procedure to solve the PIP. Instead of generating PEPs one at a time, as Dentcheva et al. do, they begin with an upper bounding vector on the PEPs as the root of a PEP search tree. They define the *level* of an integer vector as the sum of its components, and use a backwards enumeration procedure to generate PEP candidates, i.e., nodes in the search tree, one level at a time. For each candidate, they generate an integer program (IP) in which the probabilistic constraint is replaced by some lower bound on the value of any future node emanating from that branch. They then solve a linear relaxation of that IP, and are able to prune the search tree if the objective value is higher than that of the best known feasible solution. Leaves in the next level are generated for branches that are not pruned. This approach works better than that proposed by Dentcheva et al. for integer decisions variables.

In another approach to solving PIPs, Ruszczyński [2002] reformulates the probabilistic constraints as set-covering constraints, in a model that chooses which of the discrete scenarios ξ^s are to be satisfied and which are not. A partial ordering "$\preceq$" is defined on the scenarios, based on the component-wise values of their respective right-hand-side realizations: $s \preceq s' \Leftrightarrow \xi^s \leq \xi^{s'}$. If the ordering is consistent, we can generate constraints of the form $z_s \geq z_{s'}$, where $z_s = 1$ if scenario *s* is not satisfied and 0 otherwise. The PIP can be viewed as a knapsack problem, where we choose the scenarios that are not satisfied subject to a maximum aggregate probability threshold. Ruszczyński proposes a solution method that iteratively generates and lifts cover cuts until a solution to the PIP is found.

C. INITIAL FORMULATIONS

This dissertation presents work that extends research initiated in Kress, Penn and Polukarov [2004], hereafter referred to as "KPP." These authors formulate a two-level, two-period, chance-constrained inventory model (**CCIM**) with recourse, in a military setting. The KPP version of **CCIM**, which we refer to as **CCGIM** ("G" stands for "ground"), captures the basics of ground combat, in which units are assigned different sectors of a battlefront, and each must meet the opposition in its sector. We refer to the demands in each scenario as being *rigidly* assigned, because once a scenario unfolds, each combat unit is tasked with a specific mission, in its assigned sector, and the demand for ordnance induced by that mission. Each combat unit carries its own ordnance and attempts to prosecute its mission using that ordnance. Additional ordnance is stored at a central depot that can ship it to units whose supplies need to be replenished for the second period of combat.

KPP assume that the units must rely on their initial stocks to satisfy any demands in the first period (period I), and that each unit maintains sufficient safety stock with which to carry out any mission even when its nominal supply cannot. However, if any unit must use its safety stock, the scenario is considered unsatisfied. Following period I, units replenish their supplies (including their safety stocks, if used) to acceptable levels in view of the second period's (period II) projected scenarios. Replenishments are made from the central depot, or by inter-unit transfers. Thus, excess ordnance can be redistributed following period I. KPP assume the cost of capacity at the combat units is linear in the amount of supply, and this cost is factored into the price of ordnance initially allocated to the units. Hence, it is reasonable to assume that the cost of carrying ordnance with the ground forces is greater than that of storing it in the depot.

Under the conditions described above (possible consumption from the safety stock, possible transshipment during resupply, and greater cost for ordnance at the combat units than at the depot), **CCGIM** can be decomposed into two separate single-period problems and solved sequentially. The period-I problem allocates a minimum amount of ordnance so as to be able to meet uncertain demands in period I with a specified probability. The period-I allocations are used as parameters in the period-II problem that minimizes the amount of ordnance to be added to the units' inventories from the depot. Each of those problems is a special case of a combinatorial problem that KPP call the *Minmax Subset Problem* (**MSP**). KPP provide an empirically-efficient, exact algorithm to solve the **MSP** (see Kress et al. [2004]), and prove that solving the two **MSP** problems sequentially solves **CCGIM**.

We present **CCNIM** ("N" stands for "Naval"), a second variant of **CCIM** that models consumption of ASCMs in naval surface warfare. ASCMs constitute the major weapon system of modern navies that do not rely on airpower (for example, the fleets of Denmark, Greece, and the Netherlands [Baker 2002]). The naval-combat inventory model is more complex than **CCGIM**, however, and cannot be solved with KPP's simple decomposition procedure.

A key difference between ground combat and naval surface combat is the flexibility in mission assignments within a specific scenario. A high degree of interchangeability arises in meeting demands because ASCM ranges are often long compared with the distances between battling combat ships [Hughes 1995]. Consequently, once a set of targets becomes evident, the combat force has significant freedom in assigning targets to shooters. Targets are assigned among the available shooters based on available supplies and tactical positions. In some cases, each target is within range of all potential shooters; we refer to this situation as *fully flexible*. When the tactical situation does not allow every ship to engage any target, we refer to the targets in that scenario as being *semi-flexibly* assigned. In any case, we assume each target is assigned to one specific shooter, and its demand is not shared. This dissertation focuses on the fully-flexible case, which provides a lower bound (optimistic value) on the required number of missiles for any other case.

Other conditions necessary for KPP's simple decomposition approach, appropriate for ground combat, do not hold for naval combat. We assume the existence of lower and upper bounds on the number of missiles that will be carried aboard a ship because of fixed capacity constraints, operational considerations and doctrine. The assumption of a safety stock is unreasonable for combat ships, which typically have onboard inventories of at most eight ASCMs [Baker 2002]. Therefore, each ship can expend no more than its onboard inventory, and "backorders" on missiles do not arise. We assume that if the required number of missiles to cover an assigned mission exceeds the number available onboard the ship, the ship will "stay and fight" and expend its entire inventory. However, scenarios in which any ship cannot satisfy its assigned mission are considered unsatisfied. A small adjustment, which we do not pursue here, can model situations in which an insufficiently armed shooter avoids engaging its assigned target.

Once a missile is procured, we assume that the monetary costs of loading it onto a ship prior to period I or storing it at a shore-based depot are essentially equivalent, and not considered in the decision process. However, we still define a *cost ratio* between the cost of storing a missile in the depot and the cost of initially allocating a missile to a combat ship. This ratio represents our operational preferences regarding allocation strategies, by defining the relative value of missiles initially allocated to ships compared with those stored in the depot for period II. Essentially, whenever the ratio is not 1, we are willing to buy more than the minimum total number required in order to obtain an allocation plan that suits us. If we wish to reduce the operational burden of carrying out wartime replenishment operations in port, we set the depot cost higher than the on-ship cost, so the ratio exceeds one. If we wish to avoid the risk of losing missiles due to own-force casualties or expenditure on low-value targets, we can set the ratio below 1. For example, setting the cost ratio to 1.25 means that we view a solution that procures five missiles and allocates them to the ships, to be equally desirable as a solution that requires us to procure only four missiles that must be stored at the depot. Similarly, setting the cost ratio to $1+\varepsilon$, for a small $\varepsilon > 0$, corresponds to a solution that acquires the minimum total number of missiles, but allocates as many of them as possible to the ships.

Combat ships cannot normally transfer missiles directly between themselves, as ground-combat units can. That is, no direct transshipment occurs. We also assume that no ship will be <u>required</u> to make a port call for the sole purpose of offloading missiles for the use of some other ship. Therefore, the model assumes that no transshipment capability exists whatsoever, and any requirement for missiles that a ship might have following the first period of combat must be met by missiles stored in onshore depots. This assumption is conservative because, in practice, some ships will return to port for other reasons, and will be able to offload missiles. Because of the flexible assignments, lack of safety stocks, no transshipment, bounded inventories, and arbitrary cost ratio, we see that new techniques are required to solve a **CCIM** in the context of naval combat.

D. DISSERTATION OUTLINE

This dissertation presents two-period, chance-constrained models for the procurement and allocation of ASCMs for naval surface warfare. Chapter II develops **CCNIM** and its single-period variant as stochastic programs. It also presents a deterministic equivalent IP of **CCNIM**, which can be implemented in standard optimization software. Because the IP's solution times are often inadequate for practical purposes, the rest of this dissertation develops faster solution methods. Chapter III describes the properties of the "Monotonic Assignment Policy," by which a commander assigns targets with larger demands to ships with larger inventories. Because of these properties, we argue that **CCNIM** should be modified to include such a policy, and obtain **CCNIM-MAP** (**CCNIM** with constraints enforcing the Monotonic Assignment Policy). Chapter IV explores the feasible region of **CCNIM-MAP**, and identifies points within the feasible region that will be optimal if specific conditions on the objective function hold. In Chapter V, we use these observations to develop **CCNIM-dc,** a specialized decomposition algorithm, which can be implemented in a standard computer-programming language. **CCNIM-dc** provides an optimal solution for a wide range of cost ratios, and bounds the optimal cost otherwise. It requires several orders of magnitude less time than the IP to solve problems of practical size.

THIS PAGE INTENTIONALLY LEFT BLANK

II. MATHEMATICAL MODELS

This chapter describes a chance-constrained inventory model for naval surface warfare, **CCNIM**. Initially, we develop the single-period model **CCNIM-sp**, and show how it relates to the theory of probabilistic integer programs (PIPs). We then develop the full model, **CCNIM**, which spans two periods of combat. **CCNIM** allows full flexibility in mission assignment, i.e., any ship may prosecute any mission, although shooter-to-mission assignments are one-to-one. It assumes no transshipment capability, direct or indirect, between the ships, and explicitly accounts for the limited onboard inventories. We then develop the reduced fully-flexible assignment model (**RFFAM)**, a deterministic equivalent integer-programming model of **CCNIM** with improved performance compared to **FFAM**, developed in Avital [2004]. **RFFAM** is substantially faster than **FFAM,** and solves some practical instances in a reasonable length of time. However, for other instances, **RFFAM** may fail to produce even feasible solutions in one hour of computation. For this reason, we dedicate the later chapters of this dissertation to the development of specialized solution techniques for **CCNIM**.

A. CHANCE-CONSTRAINED MODELS FOR NAVAL WARFARE

1. CCNIM for a Single Period

We first develop a single-period model of the inventory problem we consider. Let n denote the number of combat ships (*shooters*) on the planner's side. Let $s \in \mathtt{S}$ denote an index set of combat *scenarios*, such that each scenario has *probability of occurrence* $\varphi_s > 0$ and $\sum_{s \in \mathtt{S}} \varphi_s = 1$. Each scenario $\mathbf{d}_s$ comprises a vector of *n demands* associated with *n missions*, which the shooters should satisfy in scenario *s*. We let $\tilde{\mathbf{d}}$ denote the *n*-dimensional, integer, random vector of demands drawn from the discrete set of scenarios $\mathtt{D} = \bigcup_{s=1}^{|\mathtt{S}|} \{\mathbf{d}_s\}$.

In practice, the vector of *n* mission demands is constructed from a conceptualized set $\mathtt{T}$ of potential *targets*. We associate a specific demand d_{ts} with target t in scenario s, which represents the number of missiles required to successfully engage the corresponding target in that scenario. Because that number is generally stochastic, we may generate several scenarios with different mission demands to represent the same target set. We set $d_{ts} = 0$ if target *t* does not actually appear in scenario *s* (it may not even be present). Let T_s denote the set of *engaged targets* in scenario *s*, defined as $\mathrm{T}_s = \{t \in \mathrm{T} \mid d_{ts} > 0\}$, and assume $|\mathrm{T}_s| \le n$ for all *s*.

Without loss of generality, we construct $\mathbf{d}_s$ by arranging components d_{ts} in such a way that $d_{ts} > 0$ for $t = 1, \ldots, |\mathrm{T}_s|$ and $d_{ts} = 0$ for $t = |\mathrm{T}_s| + 1, \ldots, \max\{n, |\mathrm{T}|\}$. Then, $\mathbf{d}_s = (d_{1s}, \ldots, d_{ns})$. Because each mission comprises at most one target, we often use the term "target" in place of "mission" in the following text.

For example, suppose that there are three shooters, and that $|\mathrm{T}| = 4$. We assume that in no scenario would the three shooters be required to engage all of the enemy targets concurrently. In scenario s_1, we encounter two of the targets, and require 2 missiles to successfully engage each one, and thus $\mathbf{d}_1 = (2, 2, 0)$. In scenario s_2, we encounter the same targets, but require two salvos to successfully engage the second target, setting $\mathbf{d}_2 = (2, 4, 0)$.

We minimize the procurement cost of missiles required to satisfy demands in projected scenarios, while satisfying a user-specified minimum probability of success p. A scenario is considered *satisfied* if all ships have enough missiles to satisfy the demands associated with their assigned missions in that scenario. The model chooses an *allocation plan* $\mathbf{v}$ that *allocates* v_i missiles to ship i and an *assignment plan* that *assigns* one mission to each ship in each scenario. Ship i's pre-combat inventory v_i is maintained between a discretionary lower bound $\underline{v}_i$ and a physical upper bound $\overline{v}_i$.

Because mission assignments are fully flexible, two scenarios are essentially equivalent if one scenario's demand vector is a permutation of the other's. For the sake of solution efficiency, if equivalent scenarios are presented as input data, they should be consolidated and appropriate adjustments made to the probability data. (Or, if an automatic scenario generator is used to create scenarios, it should be adjusted to avoid producing equivalent scenarios.) We also assume that each scenario is potentially feasible, meaning that the ships can carry the required inventories to satisfy that demand vector. If some hypothesized scenario cannot be satisfied because some of its demands exceed the capacity upper bounds, it should be excluded from the set of scenarios, and the probability threshold requirement adjusted accordingly.

The flexibility in mission assignments is expressed through the $n \times n$ permutation matrix $Y(\mathbf{v},\mathbf{d})$, belonging to the set Y of all binary matrices with a single 1 in each row and each column. In our formulation, $Y_{im}(\mathbf{v},\mathbf{d}) = 1$ if we assign mission m to ship i. Because targets are assigned flexibly, we may choose an assignment plan after the demands are observed and the existing allocations of missiles are considered. Therefore, the single-period problem is a two-stage stochastic program with recourse, in which the ship inventories $\mathbf{v}$ constitute first-stage decision variables, and the elements of $Y(\mathbf{v},\mathbf{d})$ constitute second-stage recourse variables. In practice, target assignments may not be one-to-one. However, sharing demands typically reduces the effectiveness of a missile salvo, and handling more than a single target by some ship may increase the time it requires to plan the engagement. Consequently, we require sufficient inventories to satisfy the scenarios under the one-to-one assumption.

There are several ways to explicitly formulate the optimal assignment plan $Y(\mathbf{v},\mathbf{d})$. The expression for $Y(\mathbf{v},\mathbf{d})$ in (2.4) minimizes the maximum single-target shortage, and the demand vector $\mathbf{d}$ is satisfied if $Y(\mathbf{v},\mathbf{d})$ can be found which reduces that shortage to zero. The Chance-Constrained Naval (Surface Warfare) Inventory Model for a Single Period (**CCNIM-sp**) can be expressed as:

(**CCNIM-sp**)
$$\min_{\mathbf{v}} \mathbf{1}^T \mathbf{v} \tag{2.1}$$

s.t.

$$\Pr\left\{\mathbf{v} \geq Y\left(\mathbf{v},\tilde{\mathbf{d}}\right)\tilde{\mathbf{d}}\right\} \geq p \tag{2.2}$$

$$\underline{\mathbf{v}} \leq \mathbf{v} \leq \overline{\mathbf{v}} \tag{2.3}$$

$$\text{where } Y(\mathbf{v},\mathbf{d}) \in \operatorname*{argmin}_{Y \in \mathrm{Y}} \left\{ \max_i \left\{ (Y\mathbf{d})_i - v_i \right\}^+ \right\} \tag{2.4}$$

For future reference, note that if we could not explicitly express the form of the optimal permutation $Y(\mathbf{v},\mathbf{d})$, we would treat that functional form as a decision variable, and write **CCNIM-sp** as: $\min_{\mathbf{v},Y(\cdot)} \mathbf{1}^T\mathbf{v}$, s.t. constraints (2.2) and (2.3).

Because $Y(\mathbf{v},\mathbf{d})$ represents “recourse” for the shooters, **CCNIM-sp** is not a PIP, as defined by Beraldi and Ruszczyński [2001] (see section A of Chapter I), except for some important special cases. In general, the concept of a PEP is not well defined for this problem because the optimal permutation of $\tilde{\mathbf{d}}$ depends on the number of missiles allocated to each ship and cannot be known *a priori*. An alternative formulation, where the permutations are applied to the vector of allocations $\mathbf{v}$ instead of to $\tilde{\mathbf{d}}$, is also not a PIP, in general. Although PEPs can now be found for the distribution of $\tilde{\mathbf{d}}$, the optimal solution need not lie in the union of cones emanating from the PEPs, a characteristic of a PIP. For example, suppose that the six permutations of the vector (3,2,1) are the possible scenarios, with equal probability. If we set $p = 5/6$, the only PEP is (3,3,3), but the optimal allocation plan is clearly (3,2,1), because we can assign the first ship to the target with demand 1, the second ship to the target with demand 2 and the third ship to the target with demand 3 in every scenario.

There are some important special cases where PEPs do apply. Under some assignment policies (perhaps coupled with other constraints), the matrix $Y(\mathbf{v},\mathbf{d})$ depends only on the problem data, i.e., $Y(\mathbf{v},\mathbf{d}_s) = Y(\mathbf{d}_s)\ \forall s$. In that case, we can define equivalent demand vectors and eliminate the matrix Y altogether, obtaining a PIP. For example, in **CCGIM**, where targets are rigidly assigned, we have $Y = I$, and no manipulation of the demand vectors is needed. If Y is eliminated, The PEPs of the distribution of the demand vectors are useful, and can be found, for example, by the enumeration schemes described by Beraldi and Ruszczyński [2002].

2. CCNIM

In the actual problem we wish to solve, we assume two-periods of combat, between which ships may return to port to replenish their supply from a central depot.

The index set of period-I scenarios, S^{I}, is defined as in **CCNIM-sp**. Let $s' \in \mathrm{S}^{\mathrm{II}}$ denote the index set of period-II combat scenarios. Each scenario $s' \in \mathrm{S}^{\mathrm{II}}$ has an associated conditional probability $\varphi_{s'|s}$, such that $\sum_{s' \in \mathrm{S}^{\mathrm{II}}} \varphi_{s'|s} = 1 \quad \forall s \in \mathrm{S}^{\mathrm{I}}$. Therefore, the demand vector for period II, $\tilde{\mathbf{d}}^{\mathrm{II}}$, is an integer vector whose distribution may depend on the scenario occurring in period I. We denote a realization of this conditional random vector by $\mathbf{d}^{\mathrm{II}}_{s'}$. We refer to a realization of $\left(\tilde{\mathbf{d}}^{\mathrm{I}}, \tilde{\mathbf{d}}^{\mathrm{II}}\right)$ as a "compound scenario", as defined in Chapter I. For each period-I scenario $s \in \mathrm{S}^{\mathrm{I}}$, we define the subset $\mathrm{S}^{\mathrm{II}}(s) = \left\{ s' \in \mathrm{S}^{\mathrm{II}} \mid \varphi_{s'|s} > 0 \right\}$, which indexes all possible demand vectors that might follow in period II. The subsets $\mathrm{S}^{\mathrm{II}}(s)$ may not be disjoint or even different for the various $s \in \mathrm{S}^{\mathrm{I}}$.

Let x^{I} denote the total number of missiles initially allocated to the ships, and let x^{II} denote the number of missiles initially allocated to the depot for potential use in period-II. Let v_i^{I} and v_i^{II} denote the number of missiles allocated to ship i in period I and in period II, respectively. We assume that the cost c_1 of allocating missiles to the shooters may be different from the cost c_2 of storing missiles at the depot. Initial allocation and assignment plans are chosen in period I so that the period-I scenarios are satisfied with a user-specified probability p^{I}. Following period I, **CCNIM** calculates each ship's remaining inventory r_i. Because a ship may not expend more missiles than it carries, if its assigned demand in period I exceeds its inventory, then the remaining inventory is zero. Then, for each ship i, the model supplements its remaining inventory with $w_i \geq 0$ missiles drawn from the depot, thereby setting v_i^{II}. Missions in period II are assigned so that period-II scenarios are satisfied with a user-specified probability p_s^{II}, which may depend on the realized demand scenario in period-I. **CCNIM** also ensures that each ship's inventory is maintained within required limits at the outset of each combat period. We illustrate the "flow" of missiles in Figure 1.

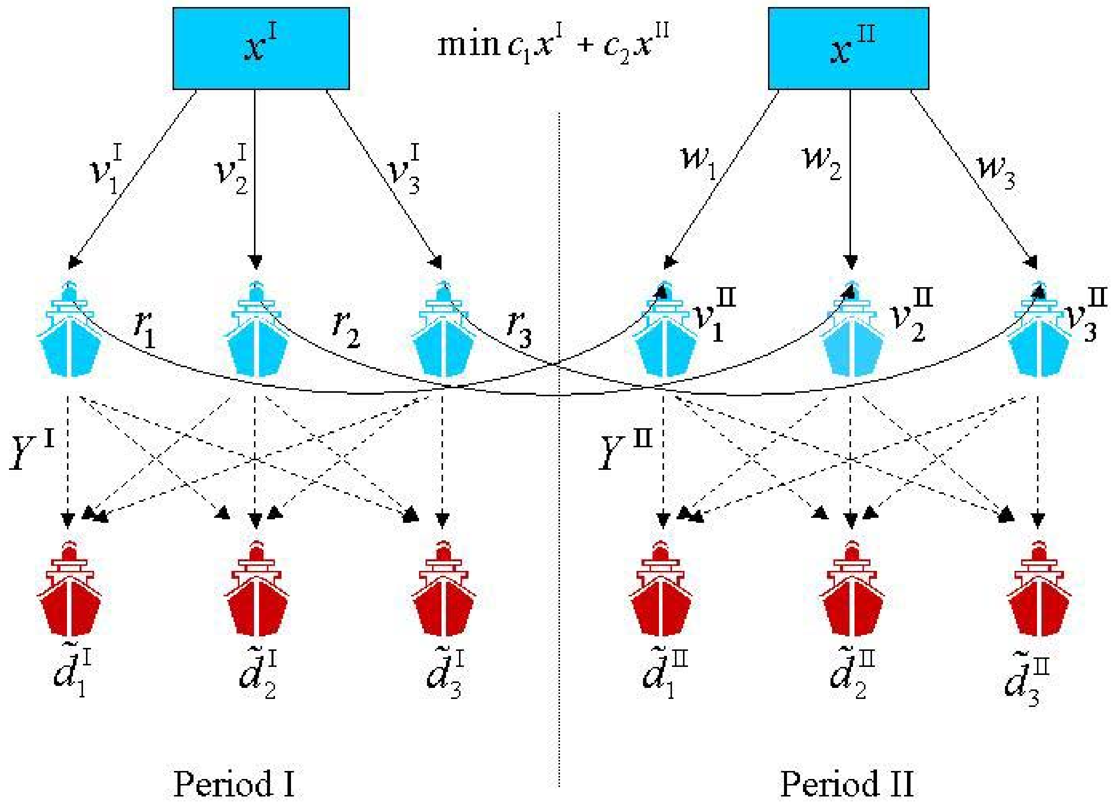

Figure 1. Schematic Missile Flow in a Compound Scenario for Three Ships. We initially procure $x^{\mathrm{I}} + x^{\mathrm{II}}$ missiles at a cost of $c_1 x^{\mathrm{I}} + c_2 x^{\mathrm{II}}$. We allocate v_i^{I} missiles to ship i, and store x^{II} in the depot. The ships face a random set of targets with a known distribution, which are represented by the demand vector $\tilde{\mathbf{d}}^{\mathrm{I}}$. The commander assigns one demand to each ship, represented by the permutation matrix Y^{I}, and, following the engagement, r_i missiles remain on ship i. We supplement ship i's inventory by w_i missiles, for a total of v_i^{II}. In period II, the ships face the random demands represented by $\tilde{\mathbf{d}}^{\mathrm{II}}$, and the commander again may choose which demand to assign to each ship, represented by Y^{II}.

Below, we formulate **CCNIM** as a chance-constrained program, which may be viewed as a three-stage stochastic program. The first stage determines the total number of missiles to procure and how many to allocate to each ship in period I. Once the period-I scenario is revealed, the model's second stage assigns missions, and, following the battle in period I, the number of missiles to add to each ship's inventory for period II.

We denote optimal second-stage decisions by $Y^{\mathrm{I}}\left(\mathbf{v},\mathbf{d},F_{\mathrm{II}}\left(\bullet|\,\mathbf{d}\right)\right)$ and $\mathbf{w}\left(\mathbf{r},F_{\mathrm{II}}\left(\bullet|\,\mathbf{d}\right)\right)$, respectively, where $F_{\mathrm{II}}\left(\bullet|\,\mathbf{d}\right)$ denotes the conditional distribution of the period-II demands. After the period II scenario is revealed, the model's third stage assigns missions for that period; we denote the optimal third stage decisions by $Y^{\mathrm{II}}\left(\mathbf{v},\mathbf{d}\right)$. However, the only "real" decision variables are the first-stage variables, after which the system is set in motion, and we simply observe whether the random demands can be covered with sufficiently high probability. The optimal second-stage and third-stage decisions are any combination of assignment plans and replenishment vectors that allow us to procure the minimum-cost package. For the reader's convenience, we summarize the notation used in **CCNIM**, followed by the model's statement:

Parameters [units]

$\tilde{\mathbf{d}}^{\mathrm{I}}$, $\tilde{\mathbf{d}}^{\mathrm{II}}$	random demand vector associated with period I and period II, respectively [missiles]
$F_{\mathrm{II}}\left(\bullet|\,\mathbf{d}\right)$	conditional distribution of period-II demands
p^{I}, p_s^{II}	probability threshold for period I, and for period II, if scenario *s* occurs in period I
c_1, c_2	unit cost of procuring a missile and allocating it to a ship or to the depot, respectively [\$/missile]
$\underline{\mathbf{v}}$, $\overline{\mathbf{v}}$	discretionary lower bound and physical upper bound on the number of missiles that may be allocated to the ships [missiles]

Decision Variables [units]

x^{I}, x^{II}	total number of missiles allocated initially to the ships and to the central depot, respectively [missiles]
$\mathbf{v}^{\mathrm{I}}$, $\mathbf{v}^{\mathrm{II}}$	numbers of missiles allocated to the ships at the outset of period I and period II, respectively [missiles]
$\mathbf{r}$	numbers of missiles left on the ships following period-I [missiles]
$\mathbf{w}\left(\mathbf{r},F_{\mathrm{II}}\left(\bullet|\,\mathbf{d}\right)\right)$	numbers of missiles receives from the depot following period-I [missiles]

$Y^{\mathrm{I}}\left(\mathbf{v},\mathbf{d},F_{\mathrm{II}}\left(\bullet\vert\,\mathbf{d}\right)\right)$ $Y^{\mathrm{II}}\left(\mathbf{v},\mathbf{d}\right)$	assignment matrix of demands to ships in period I and period II , respectively

(CCNIM)

$$\min_{x^{\mathrm{I}},x^{\mathrm{II}},\mathbf{v}^{\mathrm{I}},Y^{\mathrm{I}}(\bullet)} c_1 x^{\mathrm{I}} + c_2 x^{\mathrm{II}} \tag{2.5}$$

s.t.

$$\Pr\left\{\mathbf{v}^{\mathrm{I}} \geq Y^{\mathrm{I}}\left(\mathbf{v}^{\mathrm{I}},\tilde{\mathbf{d}}^{\mathrm{I}},F_{\mathrm{II}}\left(\bullet\vert\,\tilde{\mathbf{d}}^{\mathrm{I}}\right)\right)\tilde{\mathbf{d}}^{\mathrm{I}}\right\} \geq p^{\mathrm{I}} \tag{2.6}$$

$$\mathbf{1}^T\mathbf{v}^{\mathrm{I}} - x^{\mathrm{I}} = 0 \tag{2.7}$$

$$\underline{\mathbf{v}} \leq \mathbf{v}^{\mathrm{I}} \leq \overline{\mathbf{v}} \tag{2.8}$$

$$\mathbf{v}^{\mathrm{I}} \in \mathbb{Z}_+^n \tag{2.9}$$

$$\text{where } \mathbf{r} \equiv \left(\mathbf{v}^{\mathrm{I}} - Y^{\mathrm{I}}\left(\mathbf{v}^{\mathrm{I}},\tilde{\mathbf{d}}^{\mathrm{I}},F_{\mathrm{II}}\left(\bullet\vert\,\tilde{\mathbf{d}}^{\mathrm{I}}\right)\right)\tilde{\mathbf{d}}^{\mathrm{I}}\right)^+ \tag{2.10}$$

$$\mathbf{v}^{\mathrm{II}} \equiv \mathbf{r} + \mathbf{w}\left(\mathbf{r},F_{\mathrm{II}}\left(\bullet\vert\,\tilde{\mathbf{d}}^{\mathrm{I}}\right)\right) \tag{2.11}$$

and where $\mathbf{w}(\bullet) \in \mathbb{Z}_+^n$ is chosen such that

$$\Pr\left\{\mathbf{v}^{\mathrm{II}} \geq Y^{\mathrm{II}}\left(\mathbf{v}^{\mathrm{II}},\tilde{\mathbf{d}}^{\mathrm{II}}\right)\tilde{\mathbf{d}}^{\mathrm{II}}\right\} \geq p_s^{\mathrm{II}} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{2.12}$$

$$\mathbf{1}^T\mathbf{w}\left(\mathbf{r},F_{\mathrm{II}}\left(\bullet\vert\,\tilde{\mathbf{d}}^{\mathrm{I}}\right)\right) - x^{\mathrm{II}} \leq 0 \tag{2.13}$$

$$\underline{\mathbf{v}} - \mathbf{r} \leq \mathbf{w}\left(\mathbf{r},F_{\mathrm{II}}\left(\bullet\vert\,\tilde{\mathbf{d}}^{\mathrm{I}}\right)\right) \leq \overline{\mathbf{v}} - \mathbf{r} \tag{2.14}$$

$$\text{where } Y^{\mathrm{II}}\left(\mathbf{v},\mathbf{d}\right) \in \operatorname*{argmin}_{Y\in\Upsilon}\left\{\max_i\left\{\left(Y\mathbf{d}\right)_i - v_i\right\}^+\right\} \tag{2.15}$$

Note that **CCGIM** is a variant of this model. Because target assignments are rigid, $Y^{\mathrm{I}}\left(\mathbf{v},\mathbf{d},F_{\mathrm{II}}\left(\bullet\vert\,\mathbf{d}\right)\right) = Y^{\mathrm{II}}\left(\mathbf{v},\mathbf{d}\right) = I$. Because backorders and transshipments are allowed, r_i and w_i may be negative in some cases.

CCNIM is not a PIP for several reasons. Not only does each system of chance constraints allow permutations of the demand vector, but there are $\left|\mathbb{S}^{\mathrm{I}}\right|+1$ such systems. Even if the permutation matrix were fixed for a given missile allocation, defining relevant PEPs on the joint distribution of $\tilde{\mathbf{d}}^{\mathrm{I}}$ and $\tilde{\mathbf{d}}^{\mathrm{II}}$ would be impossible. However, we could still define PEPs separately for $\tilde{\mathbf{d}}^{\mathrm{I}}$ and for $\tilde{\mathbf{d}}^{\mathrm{II}}$ following each period-I scenario *s*, a fact we exploit in the algorithms we develop later.

B. INTEGER PROGRAMING MODELS

Stochastic programs are often converted into mixed integer programs and solved using standard optimization software, e.g. Ruszczyński [2002]. An IP representation of **CCNIM**, denoted the Fully Flexible Assignment Model (**FFAM**), is presented in Avital [2004]. **FFAM**'s computational performance is inadequate, and solution times for problems of modest size often exceed one hour. For the reader's convenience, a full description of **FFAM** is provided in Appendix A.

The following provides a formal, non-mathematical description of the Reduced Fully Flexible Assignment Model (**RFFAM**), a new IP representation of **CCNIM**. This model's formulation requires significantly fewer constraints and variables than **FFAM**, and initial results indicate it solves dramatically faster than **FFAM**. We differentiate between "regular" constraints, which are the minimum necessary to model **CCNIM**, and "specialized" constraints, which enforce the Monotonic Assignment Policy (MAP) and add valid inequalities that reduce the feasible region and improve solution times. The specialized constraints in **RFFAM** are adapted from those originally developed to improve **FFAM**'s performance. Some of the specialized constraints require data obtained from auxiliary models; we denote such data by "auxiliary parameters." A detailed explanation of the specialized constraints and the auxiliary models follows the mathematical description.

1. RFFAM Specification

The following non-mathematical description of **RFFAM** itemizes a list of problem requirements followed by the constraint keys where they are represented in the mathematical model presented in the next section.

Regular constraints

- Minimize the weighted cost of allocating missiles to the combat ships in period I and storing extra missiles at a depot for possible use in period II, (2.16).

Subject to:

- Each ship's binary allocation variable agrees with its inventory, (2.17) and (2.31).
- Each ship in each scenario is successful only if it has enough missiles to satisfy the demand of its assigned mission, (2.18) and (2.32).
- Each scenario in each period is satisfied only if every ship has enough missiles to satisfy the demand of its assigned mission, (2.19) and (2.33).
- In each period, the probability of successfully covering the scenarios exceeds a user-specified threshold, (2.20). In period II, the cumulative probability must be achieved for every possible, preceding, period-I scenario, (2.34).
- Each ship is allocated a specific number of missiles in each scenario, (2.21) and (2.35).
- Each ship in each scenario is assigned exactly one mission, (2.22) and (2.36).
- Each mission in each scenario is assigned to exactly one ship, (2.23) and (2.37).
- Each ship's inventory following the prosecution of a successful mission equals its initial level less the demand associated with its assigned mission, (2.24) and (2.25). Otherwise, the remaining inventory is zero, (2.26). (All ships "stay and fight.")
- For each ship, its remaining inventory following the prosecution of its period-I mission is a non-negative integer, bounded by the physical capacity limit (2.27).
- For each ship, the number of missiles it may carry at the outset of each combat period is bounded from below (discretionary operational constraint) and from above (physical capacity limit), (2.28), (2.29), (2.39) and (2.40).
- For each period-I scenario, each ship's total inventory of missiles, before prosecuting any period-II mission, equals the post-mission inventory after period I plus any missiles that are replenished, (2.30).
- Following each period-I scenario, the total number of missiles distributed to the ships between periods of combat may not exceed the number kept in the depot, (2.38).

Specialized constraints

- In each period-I scenario, missions with greater demands are assigned to ships with greater inventories (MAP constraints), (2.41) and (2.42).
- The number of missiles allocated to ship *i*+1 does not exceed the number allocated to ship *i*, (symmetry-breaking constraints) (2.43).
- The total number of missiles allocated in each period must exceed some minimum (total-allocation valid inequalities), (2.44), and (2.45).
- Each ship is allocated at least some minimum number of missiles in period I (single-ship valid inequalities), (2.46).

2. RFFAM Mathematical Description

Indices

$i \in I$	ships
$k \in K$	level (number) of missiles ($K = \{0, \ldots, \max \overline{v}_i\}$)
$m \in M$	missions
$s \in \mathrm{S}^{\mathrm{I}}$	scenario s in period I
$s' \in \mathrm{S}^{\mathrm{II}}$	scenario s' in period II
$\mathrm{S}^{\mathrm{II}}(s)$	Subset of period-II scenarios that may occur (with positive probability) following period-I scenario *s*

Initial Parameters [units]

d_{ms}	demand associated with mission *m* in scenario *s* [missiles]
φ_s	probability that period-I scenario *s* occurs
$\varphi_{s'\vert s}$	conditional probability that period-II scenario s' occurs, given that period-I scenario *s* occurs
p^{I}	probability threshold for period I (probability that the realized scenario must be satisfied)
p_s^{II}	probability threshold for period II, if scenario *s* occurs in period I
c_1	unit cost of procuring a missile and allocating it to a ship [\$/missile]
c_2	unit cost of procuring a missile and storing it in the depot [\$/missile]
$\underline{v}_i$	discretionary lower bound on the number of missiles that may be allocated to ship *i* [missiles]
$\overline{v}_i$	physical upper bound on the number of missiles that may be allocated to ship *i* [missiles]

Auxiliary Parameters [units]

b^{I}	minimum aggregate ship load-out required to satisfy the period-I scenarios [missiles]
b_s^{II}	minimum aggregate ship load-out required to satisfy the period-II scenarios following scenario *s* in period I [missiles]

$\underset{\smile}{v}_i$	period-I data-specific lower bound on the number of missiles that may be allocated to ship *i* [missiles]

Decision Variables [units]

(Note: As discussed later, variables v_i^{I}, v_{is}^{II} and w_{is} can be substituted out.)

v_i^{I}	number of missiles allocated to ship *i* in period I [missiles]
x_{ik}^{I}	1 if ship *i* has *k* missiles before the first combat period, and 0 otherwise
v_{is}^{II}	number of missiles allocated to ship *i* in period II following period-I scenario *s* [missiles]
x_{iks}^{II}	1 if ship *i* is replenished to level *k* missiles following period-I scenario *s*, and 0 otherwise
x^{II}	number of missiles allocated initially to the central depot for potential use in period II [missiles]
r_{is}	number of missiles left on ship *i* following period-I scenario *s* [missiles]
w_{is}	number of missiles *i* receives from the depot following period-I scenario *s* [missiles]
z_s^{I}	1 if period-I scenario *s* is satisfied, and 0 otherwise
z_{is}^{I}	1 if ship *i* successfully covers its assigned mission in period-I scenario *s* , and 0 otherwise
$z_{s's}^{\mathrm{II}}$	1 if period-II scenario *s′* is successful following period-I scenario *s*, and 0 otherwise
$z_{is's}^{\mathrm{II}}$	1 if ship *i* successfully covers its assigned mission in period-II scenario *s′* following period-I scenario *s*, and 0 otherwise
u_{ims}^{I}	1 if ship *i* is assigned mission *m* in period-I scenario *s*, and 0 otherwise
$u_{ims's}^{\mathrm{II}}$	1 if ship *i* is assigned mission *m* in period-II scenario *s′* following period-I scenario *s*, and 0 otherwise

Formulation

Due to the specialized constraints (2.41) and (2.42), constraints (2.22) and (2.23) are redundant, and remain in RFFAM for exposition purposes only.

(RFFAM) $$\min_{\mathbf{r},\mathbf{u},\mathbf{x},\mathbf{v},\mathbf{w},\mathbf{z}} c_1 \sum_i v_i^{\mathrm{I}} + c_2 x^{\mathrm{II}} \tag{2.16}$$

s. t.

Period I:

$$v_i^{\mathrm{I}} = \sum_k k x_{ik}^{\mathrm{I}} \quad \forall i \tag{2.17}$$

$$z_{is}^{\mathrm{I}} \le \frac{1}{\overline{v}_i} \left(\sum_{k \ge d_{ms}} k x_{ik}^{\mathrm{I}} - d_{ms} u_{ims}^{\mathrm{I}} \right) + 1 \quad \forall i,\, m,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.18}$$

$$z_s^{\mathrm{I}} \le z_{is}^{\mathrm{I}} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.19}$$

$$\sum_{s \in \mathbb{S}^{\mathrm{I}}} \varphi_s z_s^{\mathrm{I}} \ge p^{\mathrm{I}} \tag{2.20}$$

$$\sum_k x_{ik}^{\mathrm{I}} = 1 \quad \forall i \tag{2.21}$$

$$\sum_m u_{ims}^{\mathrm{I}} = 1 \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.22}$$

$$\sum_i u_{ims}^{\mathrm{I}} = 1 \quad \forall m,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.23}$$

$$r_{is} \le v_i^{\mathrm{I}} - \sum_m d_{ms} u_{ims}^{\mathrm{I}} + \overline{v}_i \left(1 - z_{is}^{\mathrm{I}}\right) \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.24}$$

$$r_{is} \ge v_i^{\mathrm{I}} - \sum_m d_{ms} u_{ims}^{\mathrm{I}} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.25}$$

$$r_{is} \le \overline{v}_i z_{is}^{\mathrm{I}} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.26}$$

$$r_{is} \in \{0, 1, \ldots, \overline{v}_i\}_i \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.27}$$

$$v_i^{\mathrm{I}} \in \{\underline{v}_i, \underline{v}_i + 1, \ldots, \overline{v}_i\} \quad \forall i \tag{2.28}$$

$$x_{ik}^{\mathrm{I}} \equiv 0 \quad \forall i,\, k \mid \left(k < \underline{v}_i\right) \vee \left(k > \overline{v}_i\right) \tag{2.29}$$

x_{ik}^{I}, z_{is}^{I}, z_s^{I}, u_{ims}^{I} binary.

Period II:

$$v_{is}^{\mathrm{II}} = r_{is} + w_{is} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.30}$$

$$\sum_k k x_{iks}^{\mathrm{II}} = v_{is}^{\mathrm{II}} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.31}$$

$$z_{is's}^{\mathrm{II}} \le \frac{1}{\overline{v}_i} \left(\sum_{k \ge d_{ms'}} k x_{iks}^{\mathrm{II}} - d_{ms'} u_{ims's}^{\mathrm{II}} \right) + 1 \quad \forall i,\, m,\, s \in \mathbb{S}^{\mathrm{I}},\, s \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.32}$$

$$z^{\mathrm{II}}_{s's} \le z^{\mathrm{II}}_{is's} \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.33}$$

$$\sum_{s' \in \mathbb{S}^{\mathrm{II}}(s)} \varphi_{s'|s} z^{\mathrm{II}}_{s's} \ge p^{\mathrm{II}}_s \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{2.34}$$

$$\sum_k x^{\mathrm{II}}_{iks} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.35}$$

$$\sum_m u^{\mathrm{II}}_{ims's} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.36}$$

$$\sum_i u^{\mathrm{II}}_{ims's} = 1 \quad \forall m,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.37}$$

$$\sum_i w_{is} \le x^{\mathrm{II}} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{2.38}$$

$$v^{\mathrm{II}}_{is} \in \left\{ \underline{v}_i, \underline{v}_i + 1, \ldots, \overline{v}_i \right\} \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.39}$$

$$x^{\mathrm{II}}_{iks} \equiv 0 \quad \forall i,\ k \,|\, \left(k < \underline{v}_i\right) \vee \left(k > \overline{v}_i\right),\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.40}$$

x^{II}_{iks}, $z^{\mathrm{II}}_{is's}$, $z^{\mathrm{II}}_{s's}$, $u^{\mathrm{II}}_{ims's}$ binary.

$$x^{\mathrm{II}} \in \mathbb{Z}^+ .$$

Specialized constraints:

$$u^{\mathrm{I}}_{iis} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.41}$$

$$u^{\mathrm{I}}_{ims} = 0 \quad \forall i,\ m \ne i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.42}$$

$$\sum_{k' \ge k} x^{\mathrm{I}}_{ik'} \ge \sum_{k' \ge k} x^{\mathrm{I}}_{i+1,k'} \quad \forall i \le n-1,\ k \tag{2.43}$$

$$\sum_i v^{\mathrm{I}}_i \ge b^{\mathrm{I}}. \tag{2.44}$$

$$\sum_i v^{\mathrm{II}}_{is} \ge b^{\mathrm{II}}_s \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{2.45}$$

$$x^{\mathrm{I}}_{ik} \equiv 0 \quad \forall i,\ k < \breve{v}_i \tag{2.46}$$

with these restrictions on the data:

$$\overline{v}_i \ge \overline{v}_{i+1} \ \forall i \le n-1 \tag{2.47}$$

$$d_{ms} \ge d_{m+1,s} \quad \forall m \le n-1,\ \forall s \in \mathbb{S}^{\mathrm{I}} \cup \mathbb{S}^{\mathrm{II}} . \tag{2.48}$$

This model contains some variables that can be substituted out, namely v_i^{I}, v_{is}^{II}, and w_{is}. Defining these variables in constraints (2.17), (2.31), and (2.38) generates "branching constraints," however, and branching on these variables accelerates the branch-and-bound solution process for the integer model; see Appleget and Wood [2000].

3. Explanation of the Specialized Constraints

It often happens that **RFFAM** finds an optimal integer solution fairly quickly, but the solver spends a great deal of time pruning the branch-and-bound tree before it can declare that solution optimal. The specialized constraints included in **RFFAM** help reduce solution times by substantially tightening the linear-programming (LP) lower bound, and reducing the number of nodes in the branch-and-bound tree that must be explored.

Three sets of constraints restrict target assignments to reflect the MAP (described shortly). While this restricts the problem (with respect to **CCNIM**), the optimal solution value, in our experience, seldom increases at all compared with that of the unrestricted model. Other constraints tighten the LP relaxation by enforcing lower bounds on the number of missiles that each ship requires individually and that the combat force requires as a whole to meet the demands of each combat period. These lower bounds are found by solving instances of single-period inventory models.

a. Operational-Assignment and Symmetry-Breaking Constraints

Identifying optimal assignment plans is difficult for **RFFAM**, and obviously would also be a very difficult task for a force commander to determine in real-time. (We assume a human commander assigns the targets, although an automated decision aid could also be used.) In all likelihood, the commander attempts to satisfy the current demands by following some intuitive assignment rules, and cannot assess the implications of each possible outcome on the post-battle distribution of missile inventories. If an optimal plan requires fewer missiles than one prescribed by any simple rules, then it is unlikely that the commander will be able to identify it in real-time. By using some alternative plan, he may use more missiles than the scenario prescribed, and the number of missiles in the depot may no longer be sufficient for the period-II requirements. By incorporating an assignment policy in **RFFAM**, we guarantee that the procurement levels can cover scenarios with the required probability if that policy is indeed followed.

A plausible assignment rule, which we call the *Monotonic Assignment Policy* (MAP), assigns targets with larger demands to ships with larger inventories. As well as being computationally convenient, the MAP is desirable from an operational standpoint. Assignment plans that follow the MAP minimize the maximum shortage over all targets. As a result, the MAP guarantees that all the missions are satisfied, if that is at all possible. Due to these properties, which we formally prove in Chapter III, we believe that a commander would have little motivation to choose a different assignment plan.

In principle, there can be many assignment plans (combinations of variables) that conform to the requirements of the MAP. If we assume that ships are listed in order of non-increasing upper bounds $i = 1, \ldots, n$, we can express the MAP in an efficient manner. We first reorder the demands in each scenario in non-increasing order, so that

$$d_{ms} \geq d_{m+1,s} \quad \forall m \leq n-1,\ s \in \mathbb{S}^{\mathrm{I}} \cup \mathbb{S}^{\mathrm{II}}. \tag{2.48}$$

We then assign the missions to the ships in that order for period I, obtaining the constraints

$$u^{\mathrm{I}}_{iis} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.41}$$

$$u^{\mathrm{I}}_{ims} = 0 \quad \forall i,\ m \neq i,\ s \in \mathbb{S}^{\mathrm{I}}. \tag{2.42}$$

Period-I allocations are now guaranteed to be non-increasing in the ships' index. Experience indicates that including explicit symmetry-breaking constraints,

$$\sum_{k' \geq k} x^{\mathrm{I}}_{ik'} \geq \sum_{k' \geq k} x^{\mathrm{I}}_{i+1,k'} \quad \forall i \leq n-1,\ k, \tag{2.43}$$

which force an allocation plan in which ships with a lower index are allocated more missiles than ships with a higher index, improves the model's solution time.

Similar constraints cannot be easily applied to period II because different permutations of the period-II allocation plan have different costs, which depend on the number of missiles that are added to the remaining inventory from period I. Hence, only after observing the period-I remainders, can we know specifically what inventory level to allocate to each ship.

We require complex constraints to ensure that the inventory of ship i is no less than that of ship i' if ship i' is assigned mission m and ship i is assigned any mission $m' < m$. Otherwise, there is no practical restriction on the difference between the two inventories. Such constraints can be formulated as follows, where $\overline{\overline{v}} \equiv \max_i \{\overline{v}_i\}$:

$$v_{is}^{\mathrm{II}} - v_{i's}^{\mathrm{II}} \geq \sum_{m'<m} u_{im's's}^{\mathrm{II}} \left(1+\overline{\overline{v}}\right) + \overline{\overline{v}}\left(u_{i'ms's}^{\mathrm{II}} - 2\right) - 1 \quad \forall i,\, i' \neq i,\, m \geq 2,\, s \in \mathbb{S}^{\mathrm{I}},\, s' \in \mathbb{S}^{\mathrm{II}}(s). \quad (2.49)$$

These constraints are not included in **RFFAM** because testing indicates that **RFFAM** solves faster without them. In any case, the period-II allocation in any feasible solution of **RFFAM** will successfully cover the period-II demands with probability p_s^{II} if the MAP is followed (see Corollary 1 in Chapter III). Therefore, the solution of **RFFAM** guarantees a sufficient number of missiles is procured if the MAP is followed in both periods.

b. Aggregate-Allocation Valid Inequalities

The aggregate-allocation valid inequalities (integer cutting planes) exploit the solutions of single-period problems. Recall that b^{I} is the minimum aggregate ship load-out required to satisfy the period-I scenarios and b_s^{II} is the minimum aggregate ship load-out required to satisfy the period-II scenarios following scenario *s* in period I. We can use b^{I} and b_s^{II} to generate lower bounds on the number of missiles that must be allocated to combat ships in each period. Since any feasible solution to the two-period problem must satisfy the period-I constraints, we obtain

$$\sum_i v_i^{\mathrm{I}} \geq b^{\mathrm{I}}. \quad (2.44)$$

Similarly, the total number of missiles allocated to the ships in period II must reach at least b_s^{II} following period-I scenario *s*, and from this fact we obtain

$$\sum_i v_{is}^{\mathrm{II}} \geq b_s^{\mathrm{II}} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \quad (2.45)$$

We obtain b^{I} by solving a single-period model, defined by constraints (2.16)-(2.21), (2.29), and (2.41)-(2.43), but where we set $c_1 = 1$ and $c_2 = 0$. We refer to this model as **RFFAM-sp** ("sp" stands for "single period.") We obtain b_s^{II} by solving **RFFAM-spII** ("spII" stands for "single period on period II"), a similar model defined on the period-II scenarios $s' \in \mathbb{S}^{\mathrm{II}}(s)$ with their respective demands, probabilities and specified probability thresholds. **RFFAM-spII** comprises the following constraints from **RFFAM**: (2.31)-(2.37) and (2.39)-(2.40), but requires the modified objective function:

$$\min_{\mathbf{u},\mathbf{x},\mathbf{v},\mathbf{z}} \sum_{s \in \mathbb{S}^{\mathrm{I}}} \sum_i v_{is}^{\mathrm{II}} . \tag{2.50}$$

We also include a version of the MAP, to accelerate solution of the model:

$$u_{iis's}^{\mathrm{II}} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) . \tag{2.51}$$

The explicit formulation of both models appears in Appendix B.

Because of the MAP, **RFFAM-sp** and **RFFAM-spII** are both PIPs (**RFFAM-spII** must first be decomposed into $|\mathbb{S}^{\mathrm{I}}|$ problems, each with a single probabilistic constraint) and could be solved by enumerating PEPs, by a specialized method utilizing PEPs, or with the **MSP** algorithm of KPP. However, for problems of our size, they can be solved as IPs by branch and bound. The computation time for both models is negligible compared to that of **RFFAM** for problems requiring more than a few seconds to solve.

c. Single-Ship Valid Inequalities

The discretionary lower bound on a single ship's inventory, set in constraints (2.28), (2.29), (2.39) and (2.40), derives from generic operational considerations. Tighter constraints can be derived from specific problem data.

We define a modified single-period model denoted **RFFAM-lb** ("lb" stands for "lower bound") in which we require the set of $n|\mathbb{S}^{\mathrm{I}}|$ missions to be assigned to the ships as before. This model uses the same constraints as **RFFAM-sp**, except that we drop constraints (2.19) and replace z_s by z_{is} in constraint (2.20), so that each ship selects its set of successful scenarios independently of the other ships (although mission assignments are not independent.) This relaxes **RFFAM-sp** because each ship can reach the specified probability threshold p^{I} by satisfying demands from a set of scenarios different from those of another ship.

We solve **RFFAM-lb** and obtain an optimal allocation plan $\underset{\smile}{\mathbf{v}} = \{\underset{\smile}{v}_1, \cdots, \underset{\smile}{v}_n\}$ that defines the number of missiles *k* allocated to each ship (as specified for **RFFAM**). We then modify the lower bounds on the capacities by setting

$$x_{ik}^{\mathrm{I}} \equiv 0 \quad \forall i,\ k < \underset{\smile}{v}_i \tag{2.46}$$

before solving **RFFAM**.

These inequalities are valid because constraints (2.41) and (2.42) together completely control the period-I target assignment plans. Therefore, the target-to-shooter assignment plans generated by **RFFAM-lb** are identical to those generated by **RFFAM** in period I. The only open decision actually left to **RFFAM** in period I and to **RFFAM-lb** is the choice of an allocation plan with a sufficient number of missiles to meet the probability threshold.

Because the target assignments are forced through the assignment rules, **RFFAM-lb** is a PIP. In fact, the solution of **RFFAM-lb** provides the same lower bound on the level of PEPs for **RFFAM-sp** as would have been obtained by using the method described in Dentcheva et al. [2000]. There, the lower bounding vector on any PEP is obtained by using the marginal distribution for each component of the demand vector.

Equivalent constraints cannot be formulated for period II, because the remainders are not known in advance. For example, consider the two period-II demand vectors (6,3,1) and (5,3,3), and assume that only one of them must be satisfied. Then solving a single period model with this data produces the optimal allocation (6,3,1). However, if the three ships have 3 missiles remaining each, following the period-I scenario, then it would be less costly to choose to satisfy the scenario (5,3,3), and replenish 2 missiles from the depot, rather than satisfying the scenario (6,3,1), which requires 3 additional missiles. If (6,3,1) were used as single ship lower bounds in period II, the minimum allocation would be (6,3,3), which is excessive.

C. SUMMARY OF COMPUTATIONAL BEHAVIOR

We compare the solution times of **RFFAM** to those of **FFAM** using the CPLEX solver version 9.0 [ILOG 2003] on a Pentium IV, 2 GHz personal computer with 1 Gbyte of RAM. We set all cost ratios using the smallest integers possible, and set the relative optimality criterion to 0.0%, but the absolute optimality criterion to 0.99.

We randomly generate scenarios by drawing demands from a discrete uniform distribution between 0 and 8 (details on test cases appear in Appendix C). All of the scenarios are equi-probable, and the probability threshold in each period requires us to satisfy all but one scenario in each case. We refer to a complete specification of the parameters required to solve **FFAM** as an *instance*. We generate several instances from each case specification by varying the cost ratio. Not only does this change the objective function and result in different optimal solutions, but the computational effort required to solve the different instances changes as well. We obtain the parameters required for the specialized constraints by solving one instance each of **RFFAM-sp**, **RFFAM-spII**, and **RFFAM-lb**. The auxiliary models solve very quickly, so we ignore this computational effort when presenting results.

Table 1 summarizes the computation time of both **FFAM** and **RFFAM** for the four cases examined. For each case, we list the number of ships involved, and number of period-I and period-II scenarios. For each instance, we report the solution time, in seconds, for **FFAM** and **RFFAM**.

The table shows that **FFAM**'s run-time performance is too slow for many practical instances, which may have as many as fifteen ships, and often more than ten scenarios in each period. **RFFAM** outperforms **FFAM** in every one of the instances presented, usually by several orders of magnitude. In fact, **RFFAM** solves every instance but one in less than 3 seconds. In that instance (case 2c, with cost ratio 0.5) a solution within 3% optimal is found in less than one second.

		(c_1, c_2)					
Case Name	$(n, \lvert S^{I} \rvert, \lvert S^{II} \rvert)$	(2,1)		(1,1)		(1,2)	
		FFAM	**RFFAM**	**FFAM**	**RFFAM**	**FFAM**	**RFFAM**
Case 2a	(3,3,3)	4.437	0.359	21.42	0.249	11.47	0.249
Case 2b	(4,4,4)	393.6	0.656	111.6	1.453	146.3	0.328
Case 2c	(5,5,5)	>3600[a]	98.59	>3600[b]	2.812	730.8	2.046
Case 2d	(6,6,6)	2511.2	0.874	>3600[c]	1.374	2839.1	1.473

Table 1. Comparison of **FFAM** and **RFFAM** Computation Times.

The table lists the amount of time, in seconds, required by CPLEX, version 9.0, to solve various instances of **FFAM** and **RFFAM**. The top row lists the cost coefficients used. Each subsequent row lists the times required for that instance. Case size is specified by three parameters, $(n, \lvert S^{I} \rvert, \lvert S^{II} \rvert)$. Each scenario is equi-probable, and one scenario in each period may be unsatisfied. Instances that do not solve in an hour are indicated by >3600, and the remaining relative gaps are as follows:
a) 4.23%. b) 2.22%. c) 6.28%.

Although **RFFAM** is much faster than **FFAM**, its run times are less predictable than **FFAM**'s and may become excessive. As is apparent from the table, **RFFAM** requires more time to solve the instances of case 2c than those of case 2d for equivalent cost ratios, although case 2c has fewer ships and scenarios. In fact, instances of comparable size can be found, for which the solver is unable to report an optimal solution following one hour of computation (case 2e with costs (2,1)). In a larger instance, involving 8 ships and 8 scenario in each period (case 2f with costs (1, 1)), the solver fails to even identify any feasible integer solution in the allotted time. Ultimately, we believe **RFFAM** is not a reliable option for solving practical instances of **CCNIM**.

D. CONCLUSIONS

This chapter has developed **CCNIM**, the two-period chance-constrained inventory model for naval surface warfare. The model is a relatively complex three-stage stochastic program. We also formulate the deterministic-equivalent **RFFAM**, an improved formulation over **FFAM**, which was developed in Avital [2004]. Solution times for **RFFAM** are substantially better than for **FFAM**, but neither model can reliably solve problems of practical size. In the following chapters, we develop alternative, faster solution techniques.

III. THE MONOTONIC ASSIGNMENT POLICY

We introduced the MAP in Chapter II, which is the mission allocation policy that assigns targets with larger demands to ships with larger inventories, in part, to improve the solution times for the IP models. More important than improving solution times, we believe the MAP is a reasonable approximation of how actual assignments should and would be carried out because assigning targets according to the MAP guarantees that all the missions are satisfied, if that is at all possible. Furthermore, the MAP guarantees the lowest cost solution to any single-period problem, so it is also logistically efficient. For these reasons, we believe that the MAP should be implemented in any algorithm that calculates a procurement and allocation plan for naval surface warfare, and formulate **CCNIM-MAP,** a restriction of **CCNIM** that incorporates MAP constraints.

A. OPERATIONAL PROPERTIES OF THE MAP

We identify three properties of a MAP, which hold for any given allocation.

Property 3.1: A MAP minimizes the maximum shortage over all targets in any scenario.

Property 3.2: A MAP satisfies any single-period scenario that can be satisfied by a given set of inventories.

Property 3.3: A MAP expends the greatest number of missiles in any scenario (a desirable characteristic when planning procurement, as we explain later).

We proceed to prove these properties in several lemmas, below.

Because each property depends only on the relation of a set of ships' inventories to the ships' assigned missions, we assume, without loss of generality, that the inventories are non-increasing, i.e., $v_1 \geq v_2 \geq \cdots \geq v_n$, for ships $i = 1, \ldots, n$. Let $\mathbf{d} = (d_1, ..., d_n) \in \mathbb{Z}_+^n$ represent the demands of an arbitrary scenario. There are $n!$ permutations of the elements of $\mathbf{d}$. Let $\mathrm{K} = \{1, ..., n!\}$ be the index set of all permutations, and let $\mathbf{d}^k$, $k \in \mathrm{K}$, denote the k^{th} permutation of $\mathbf{d}$. We denote the monotonically ordered demand vector by $\mathbf{d}^*$, and let k^* denote some corresponding permutation index. Note that $\mathbf{d}^*$ is unique, although, due to ties in the data, there may be several corresponding permutation indices. We represent alternative assignment plans by different permutations of the demand vector, indexed by *k*, and assign demand d_i^k to ship i.

Lemma 1.

Let $\mathbf{v} \in \mathbb{Z}_+^n$ be a non-increasing vector. Let $\mathbf{d} \in \mathbb{Z}_+^n$ be a vector of demands, and let $\mathbf{d}^*$ be a permutation of $\mathbf{d}$ whose components are non-increasing. Let $\delta^k = \max_i \left\{ d_i^k - v_i \right\}^+$ denote the greatest single-target shortage associated with the assignment plan represented by $\mathbf{d}^k$. Then $\delta^{k^*} = \min_{k \in \mathrm{K}} \delta^k$.

Proof:

Let $k \in \operatorname{argmin}_{k \in \mathrm{K}} \delta^k$, and suppose that $\mathbf{d}^k \neq \mathbf{d}^*$. Then, there exist demands *l* and *m* such that $l < m$ and $d_l^k < d_m^k$. Consider $\mathbf{d}^{k'}$, an alternative permutation of $\mathbf{d}$, in which

$$d_m^{k'} = d_l^k \tag{3.1}$$

$$d_l^{k'} = d_m^k \tag{3.2}$$

$$d_i^{k'} = d_i^k \;\; \forall i \neq l, m. \tag{3.3}$$

Let $\varepsilon^k(l,m) = \max_{i \neq l,m} \left\{ d_i^k - v_i \right\}^+$. Then, $\delta^k = \max\left\{ \varepsilon^k(l,m),\, d_l^k - v_l,\, d_m^k - v_m \right\}$ and $\delta^{k'} = \max\left\{ \varepsilon^k(l,m),\, d_m^k - v_l,\, d_l^k - v_m \right\}$. By our assumptions, we have $d_m^k - v_l \leq d_m^k - v_m$ and $d_l^k - v_m < d_m^k - v_m$, so $\delta^{k'} \leq \delta^k$.

If $\mathbf{d}^{k'} \neq \mathbf{d}^*$, set $k = k'$, and find a new permutation $\mathbf{d}^{k'}$ that satisfies conditions (3.1)-(3.3). Because $|\mathrm{K}|$ is finite and each k' indexes a different permutation of $\mathbf{d}$, the sequence of permutation indices $k, k', \ldots, k^{(\kappa)}$ must reach $\mathbf{d}^*$ without increasing δ^k. ∎

The proof of the following corollary is then obvious.

Corollary 1.

Let $\mathbf{v} \in \mathbb{Z}_+^n$ be a non-increasing vector. Let $\mathbf{d} \in \mathbb{Z}_+^n$ be a vector of demands, and let $\mathbf{d}^*$ be a permutation of $\mathbf{d}$ whose components are non-increasing. If $\mathbf{v} \geq \mathbf{d}^k$ for some permutation index *k*, then $\mathbf{v} \geq \mathbf{d}^*$. ∎

Lemma 1 formalizes property 3.1, i.e., that a monotonic plan always minimizes the maximum single-target shortage. In particular, that shortage is zero in satisfied scenarios. This leads directly to Corollary 1, which implies that if any successful assignment plan exists, then a monotonic plan is successful as well.

As an example of Lemma 1, consider a situation with three targets, corresponding demand vector $\mathbf{d} = (4,4,2)$, and an inventory vector for three shooters of $\mathbf{v} = (4,3,1)$. Under a MAP, the ith target is assigned to the ith ship, $i = 1, 2, 3$. The maximum single-target deficiency is one missile. If the commander were to switch the last two assignments, the second ship would not use its entire inventory, but the second target would be engaged with only a single missile, increasing the maximum deficiency to three missiles.

Finally, Lemma 2 proves property 3.3, i.e., that adhering to the MAP guarantees expending at least as many missiles in any scenario as any other assignment plan, without engaging any target with more missiles than it requires, of course. As a result, if a different assignment plan is chosen for any reason (for example, when the assumption of full-flexibility does not hold), the total number of missiles left over from period I does not decrease. However, the number of remaining missiles on each ship may be different following an alternative assignment. Therefore, some ships may now carry more missiles than they require in period II, while other ships need to replenish more missiles than originally planned. Because of the no-transshipment policy, the depot inventory may now be insufficient for period II. However, by recalling some “overstocked” ships to port and violating the no-transshipment policy, we are guaranteed to have enough missiles to meet the period-II required allocations. This violation may occur naturally, because some overstocked ships may be recalled to port for other purposes. Although property 3.3 does not directly affect the efficiency of any algorithm, it implies additional robustness for a procurement solution provided by **CCNIM**, and further motivates the use of a MAP.

Lemma 2.

Let $\mathbf{v} \in \mathbb{Z}_+^n$ be a non-increasing vector. Let $\mathbf{d} \in \mathbb{Z}_+^n$ be a vector of demands, and let $\mathbf{d}^*$ be a permutation of $\mathbf{d}$ whose components are non-increasing. Let $\eta_i^k = \min\left\{v_i, d_i^k\right\}$ be the number of missiles fired by ship i associated with the assignment plan represented by $\mathbf{d}^k$. Then $\sum_i \eta_i^{k^*} = \max_{k \in \mathrm{K}} \sum_i \eta_i^k$.

Proof:

Let $k \in \operatorname{argmax}_{k' \in \mathrm{K}} \sum_i \eta_i^{k'}$, and suppose that $\mathbf{d}^k \neq \mathbf{d}^*$. Then, there exist demands *l* and *m* such that $l < m$ and $d_l^k < d_m^k$. Consider $\mathbf{d}^{k'}$, an alternative permutation of $\mathbf{d}$ that satisfies conditions (3.1)-(3.3). We obtain

$$\sum_i \eta_i^{k'} - \sum_i \eta_i^k = \left[\min\left\{v_l, d_l^{k'}\right\} + \min\left\{v_m, d_m^{k'}\right\}\right] - \left[\min\left\{v_l, d_l^k\right\} + \min\left\{v_m, d_m^k\right\}\right]. \quad (3.4)$$

Because we assume $v_l \geq v_m$ and $d_l^k < d_m^k$, there are six cases to examine:

- If $v_m \geq d_m^k$, then equation (3.4) reduces to $\left[d_m^k + d_l^k\right] - \left[d_l^k + d_m^k\right] = 0$.
- If $v_l \leq d_l^k$, then we are left with $\left[v_l + v_m\right] - \left[v_l + v_m\right] = 0$.
- If $v_l \geq d_m^k \geq v_m \geq d_l^k$, we have $\left[d_m^k + d_l^k\right] - \left[d_l^k + v_m\right] = d_m^k - v_m \geq 0$.
- If $v_l \geq d_m^k > d_l^k \geq v_m$, we have $\left[d_m^k + v_m\right] - \left[d_l^k + v_m\right] = d_m^k - d_l^k > 0$.
- If $d_m^k \geq v_l \geq d_l^k \geq v_m$, we have $\left[v_l + v_m\right] - \left[d_l^k + v_m\right] = v_l - d_l^k \geq 0$.
- If $d_m^k \geq v_l \geq v_m \geq d_l^k$, we have $\left[v_l + d_l^k\right] - \left[d_l^k + v_m\right] = v_l - v_m \geq 0$.

If $\mathbf{d}^{k'} \neq \mathbf{d}^*$, set $k = k'$, and find a new permutation $\mathbf{d}^{k'}$ that satisfies conditions (3.1)-(3.3). Because $|\mathrm{K}|$ is finite and each k' indexes a different permutation of $\mathbf{d}$, the sequence of permutation indices $k, k', ..., k^{(\kappa)}$ must reach $\mathbf{d}^*$ without decreasing $\sum_i \eta_i^k$. ∎

B. OPTIMAL SOLUTION TO CCNIM-sp

This section proves that, assuming the shooters can be listed such that $\underline{\mathbf{v}}$ and $\overline{\mathbf{v}}$ are both non-increasing, then assigning targets according to the MAP guarantees an optimal solution to **CCNIM-sp**. We believe this assumption is reasonable in practical situations, as will be discussed below. The optimality of the MAP in single-period problems is significant because, by applying the MAP, **CCNIM-sp** is reduced to a PIP. The specialized algorithm for solving **CCNIM**, which we develop later in this dissertation, can then take advantage of the properties of PIPs to solve problems quickly.

To prove that assigning targets according to the MAP is optimal for instances of **CCNIM-sp**, we first prove that the MAP is optimal for the *Flexible Minmax Subset Problem* (**FMSP**). **FMSP** is a combinatorial problem that is based on the *Minmax Subset Problem* (**MSP**) defined by KPP [2004]. We obtain **FMSP** by eliminating the capacity constraints (2.3) in **CCNIM-sp**.

Let $\mathbf{d}_s = (d_{1s}, ..., d_{ns}) \in \mathbb{Z}_+^n$ denote a demand vector for scenario $s \in \mathrm{S}$ with associated probability $\varphi_s > 0$. Let $\mathrm{K}_s = \{1, ..., n!\}$ be the index set of all permutations of $\mathbf{d}_s$, and let $\mathbf{d}_s^{k_s}$, $k_s \in \mathrm{K}_s$, denote the vector induced by the kth permutation of $\mathbf{d}_s$. Let $\mathbf{k} = (k_1, ..., k_{|\mathrm{S}|})$ denote a vector of permutation indices, one for each scenario, which we refer to as an *arrangement indicator*. Let $\mathrm{D}^{\mathbf{k}} = \bigcup_s \{\mathbf{d}_s^{k_s}\}$, and let $\mathrm{A} = \bigcup \mathbf{k}$. Note that $|\mathrm{A}| = (n!)^{|\mathrm{S}|}$.

A subset of scenario indices $\mathrm{S}_f \subseteq \mathrm{S}$ is said to be *p-feasible* for a given probability parameter *p* if $\sum_{s \in \mathrm{S}_f} \varphi_s \geq p$. Let $\mathrm{S}_\mathrm{F} = \left\{ \mathrm{S}_f \mid \sum_{s \in \mathrm{S}_f} \varphi_s \geq p \right\}$ denote the set of *p*-feasible subsets.

Let $\bar{d}_i^{\mathbf{k}}\left(\mathtt{S}_f\right) = \max_{s\in\mathtt{S}_f} d_{is}^{k_s}$; the vector $\bar{\mathbf{d}}^{\mathbf{k}}$ exactly prescribes the largest demand that each ship will need to satisfy in the scenarios that comprise the *p*-feasible subset. Also, define $D\left(\mathtt{S}_f,\mathbf{k}\right) = \sum_{i=1}^{n} \bar{d}_i^{\mathbf{k}}\left(\mathtt{S}_f\right)$. We now define **FMSP** precisely:

Definition: **FMSP** (*Flexible Minmax Subset Problem*): Find a *p*-feasible subset $\underline{\mathtt{S}}$ and associated arrangement indicator $\underline{\mathbf{k}}$ such that $D\left(\underline{\mathtt{S}},\underline{\mathbf{k}}\right) = \min_{\mathtt{S}_f\in\mathtt{S}_\mathrm{F},\mathbf{k}} D\left(\mathtt{S}_f,\mathbf{k}\right)$; we refer to $\underline{\mathtt{S}}$ as a "minimal *p*-feasible subset." ∎

We may also formulate **FMSP** as:

$$D\left(\underline{\mathtt{S}},\underline{\mathbf{k}}\right) = \min_{\mathtt{S}_f\in\mathtt{S}_\mathrm{F},\mathbf{k}} \sum_{i=1}^{n} \max_{s\in\mathtt{S}_f} d_{is}^{k_s} \tag{3.5}$$

$$\text{s.t.} \qquad \sum_{s\in\mathtt{S}_f} \varphi_s \geq p$$

Let $\mathtt{A}_\mathrm{M} = \left\{\mathbf{k} \mid d_{is}^{k_s} \geq d_{js}^{k_s}\ \forall i<j,\ \forall s\in\mathtt{S}\right\}$ be the set of arrangement indicators where each vector $\mathbf{d}_s^{k_s}$ is non-increasing. Because monotonic orderings are unique to within ties in the data, the vectors $\mathbf{d}_s^{k_s}$ and their upper-bounding vector $\bar{\mathbf{d}}^{\mathbf{k}}$ are unique, regardless which $\mathbf{k}\in\mathtt{A}_\mathrm{M}$ is used. We denote these vectors by $\mathbf{d}_s^*$ and $\bar{\mathbf{d}}^*$ respectively, by $\mathtt{D}^*$ the arrangement of $\mathbf{d}_s^*$, and by $\mathbf{k}^*$ some $\mathbf{k}\in\mathtt{A}_\mathrm{M}$. (If the data exhibit no ties, then $\mathtt{A}_\mathrm{M} = \left\{\mathbf{k}^*\right\}$.) Furthermore, let $\mathtt{A}_\mathrm{MM} = \left\{\mathbf{k} \mid \bar{d}_i^{\mathbf{k}} \geq \bar{d}_j^{\mathbf{k}}\ \forall i<j\right\}$ be the set of arrangement indicators for which the elements of the resulting vector of maxima $\bar{\mathbf{d}}^{\mathbf{k}}$ form a non-increasing sequence ("MM" stands for "monotonic maxima"). For notational convenience, we drop the set argument if $\bar{d}_i^{\mathbf{k}}\left(\mathtt{S}\right)$ is calculated over the entire set $\mathtt{S}$. Clearly, $\mathtt{A}_\mathrm{M} \subseteq \mathtt{A}_\mathrm{MM} \subseteq \mathtt{A}$.

We proceed to prove that solving **FMSP** using a fixed arrangement $\mathtt{D}^*$, corresponding to a monotonic plan, yields the same optimal value as an unrestricted solution of **FMSP**, i.e., $\min_{\mathtt{S}_f\in\mathtt{S}_\mathrm{F}} D\left(\mathtt{S}_f,\mathbf{k}^*\right) = \min_{\mathtt{S}_f\in\mathtt{S}_\mathrm{F},\mathbf{k}} D\left(\mathtt{S}_f,\mathbf{k}\right)$.

Lemma 3.

Let $\mathbb{D} = \bigcup_s \{\mathbf{d}_s\}$ be a set of demand vectors, and let $\mathbb{D}^*$ be an arrangement of those vectors in which $\mathbf{d}_s^*$ is non-increasing for all s. Recall that $\bar{d}_i^{\mathbf{k}} = \max_{s\in\mathbb{S}} d_i^{k_s}$, and let $\bar{d}_i^* = \max_{s\in\mathbb{S}} d_i^*$. Then $\bar{d}_i^* = \min_{\mathbf{k}\in\mathbb{A}_{\mathrm{MM}}} \bar{d}_i^{\mathbf{k}}\ \forall i$.

Proof:

We prove the lemma by showing that there is a sequence of arrangement indicators leading from any $\mathbf{k}\in\mathbb{A}_{\mathrm{MM}}$ to some $\mathbf{k}^{(\kappa)}\in\mathbb{A}_{\mathrm{M}}$, where each successive arrangement, obtained by interchanging elements in a permutation of a single scenario, maintains the relation $\bar{\mathbf{d}}^{\mathbf{k}} \geq \cdots \geq \bar{\mathbf{d}}^{\mathbf{k}^{(\kappa)}}$.

Let $\mathbf{k}\in \operatorname{argmin}_{\mathbf{k}\in\mathbb{A}_{\mathrm{MM}}} \bar{d}_i^{\mathbf{k}}$, and suppose that $\mathbf{k}\notin\mathbb{A}_{\mathrm{M}}$. Then, there exist scenario t and demands l and m with $l<m$ and $d_{lt}^{k_t} < d_{mt}^{k_t}$. Consider $\mathbb{D}^{\mathbf{k}'}$, an alternative arrangement of $\mathbb{D}^{\mathbf{k}}$, in which

$$d_{mt}^{k_t'} = d_{lt}^{k_t} \tag{3.6}$$

$$d_{lt}^{k_t'} = d_{mt}^{k_t} \tag{3.7}$$

$$d_{is}^{k_s'} = d_{is}^{k_s}\ \ \forall i\neq l,m \text{ if } s=t \text{ and } \forall i \text{ if } s\neq t. \tag{3.8}$$

Let $\bar{\mathbb{S}}_t = \mathbb{S}-\{t\}$, and note that $\bar{d}_i^{\mathbf{k}} = \max\left\{\bar{d}_i^{\mathbf{k}}\left(\bar{\mathbb{S}}_t\right),\, d_{it}^{k_t}\right\}\ \forall i$. By our assumptions, $\bar{d}_l^{\mathbf{k}} \geq \bar{d}_m^{\mathbf{k}}$ and $d_{lt}^{k_t} < d_{mt}^{k_t}$. Because $\bar{d}_m^{\mathbf{k}} \geq d_{mt}^{k_t}$, we obtain $\bar{d}_l^{\mathbf{k}} = \bar{d}_l^{\mathbf{k}}(\bar{\mathbb{S}}_t) > d_{lt}^{k_t}$ and $\bar{d}_l^{\mathbf{k}} \geq d_{mt}^{k_t}$. Therefore, $\bar{d}_l^{\mathbf{k}'} = \max\left\{\bar{d}_l^{\mathbf{k}}\left(\bar{\mathbb{S}}_t\right),\, d_{mt}^{k_t}\right\} = \bar{d}_l^{\mathbf{k}}\left(\bar{\mathbb{S}}_t\right) = \bar{d}_l^{\mathbf{k}}$. Because $d_{mt}^{k_t'} < d_{mt}^{k_t}$, we get $\bar{d}_m^{\mathbf{k}'} \leq \bar{d}_m^{\mathbf{k}}$, and, considering that $\bar{d}_i^{\mathbf{k}'} = \bar{d}_i^{\mathbf{k}}\ \forall i\neq\{l,m\}$, we obtain $\bar{d}_i^{\mathbf{k}'} \leq \bar{d}_i^{\mathbf{k}}\ \forall i$.

Because $\bar{d}_m^{\mathbf{k}'} \leq \bar{d}_m^{\mathbf{k}}$, $\bar{\mathbf{d}}^{\mathbf{k}'}$ may no longer be monotonic and it may be true that $\mathbf{k}'\notin\mathbb{A}_{\mathrm{MM}}$. Let $j = \operatorname{argmax}_{i>m}\left\{\bar{d}_m^{\mathbf{k}'} < \bar{d}_i^{\mathbf{k}'}\right\}$. If $j\neq\varnothing$, let the arrangement indicator $\mathbf{k}^1$ be such that $d_{is}^{k_s^1} = d_{i+1,s}^{k_s'}\ \ \forall m\leq i\leq j-1,\ \forall s$, $d_{js}^{k_s^1} = d_{ms}^{k_s'}\ \ \forall s$, and $d_{is}^{k_s^1} = d_{is}^{k_s'}\ \ \forall i<m \vee i>j,\ \forall s$.

Thus, we are effectively permuting the entries of $\boldsymbol{\mu}^{\mathbf{k}'}$ without changing their values. Because $\bar{d}_i^{\mathbf{k}'} \geq \bar{d}_{i+1}^{\mathbf{k}'}\ \forall i > m$ and $\bar{d}_j^{\mathbf{k}'} \geq \bar{d}_m^{\mathbf{k}'}$, we have $\mathbf{k}^1 \in \mathbb{A}_{\text{MM}}$, and set $\mathbf{k}' = \mathbf{k}^1$.

If $\mathbf{k}' \notin \mathbb{A}_{\text{M}}$, set $\mathbf{k} = \mathbf{k}'$, and find a new arrangement $\mathbb{D}^{\mathbf{k}'}$ that satisfies conditions (3.6)-(3.8). Because $|\mathbb{A}_{\text{MM}}|$ is finite and each $\mathbf{k}'$ leads to a different member of $\mathbb{A}_{\text{MM}}$ (possibly through $\mathbf{k}^1$) the sequence of arrangement indicators $\mathbf{k}, \mathbf{k}', \ldots, \mathbf{k}^{(\kappa)}$ must reach some $\mathbf{k}^{(\kappa)} \in \mathbb{A}_{\text{M}}$, and $\bar{\mathbf{d}}^{\mathbf{k}^{(\kappa)}} = \bar{\mathbf{d}}^*$ by definition. ∎

The proof of the following corollary is then obvious.

Corollary 2.

Let $D(\mathbb{S}, \mathbf{k}) = \sum_{i=1}^{n} \bar{d}_i^{\mathbf{k}}$. Then $D(\mathbb{S}, \mathbf{k}^*) = \min_{\mathbf{k} \in \mathbb{A}_{\text{MM}}} D(\mathbb{S}, \mathbf{k})$. ∎

Lemma 4.

For a fixed set of vectors $\mathbb{D}$ and corresponding index set $\mathbb{S}$, the arrangement $\mathbb{D}^* = \bigcup_s \{\mathbf{d}_s^*\}$ minimizes the sum of component-wise maxima over all arrangements, i.e. $D(\mathbb{S}, \mathbf{k}^*) = \min_{\mathbf{k} \in \mathbb{A}} D(\mathbb{S}, \mathbf{k})$.

Proof:

Let $\mathbf{k} \in \operatorname{argmin}_{\mathbf{k} \in \mathbb{A}} D(\mathbb{S}, \mathbf{k})$, so $D(\mathbb{S}, \mathbf{k}) \leq D(\mathbb{S}, \mathbf{k}^*)$. If $\mathbf{k} \notin \mathbb{A}_{\text{MM}}$, there must be indices l and m with $l < m$ and $\bar{d}_l^{\mathbf{k}} < \bar{d}_m^{\mathbf{k}}$. Consider $\mathbb{D}^{\mathbf{k}'}$, an alternative arrangement of $\mathbb{D}$, in which

$$d_{ls}^{k_s'} = d_{ms}^{k_s} \quad \forall s \tag{3.9}$$

$$d_{ms}^{k_s'} = d_{ls}^{k_s} \quad \forall s \tag{3.10}$$

$$d_{is}^{k_s'} = d_{is}^{k_s} \quad \forall i \neq l, m\ \forall s. \tag{3.11}$$

In effect, the arrangement $\mathtt{D}^{\mathbf{k}'}$ switches two components of $\bar{\mathbf{d}}^{\mathbf{k}}$ so that $\bar{d}_l^{\mathbf{k}'} = \bar{d}_m^{\mathbf{k}}$, $\bar{d}_m^{\mathbf{k}'} = \bar{d}_l^{\mathbf{k}}$ and $\bar{d}_i^{\mathbf{k}'} = \bar{d}_i^{\mathbf{k}}$ $\forall i \neq \{l, m\}$. Obviously, the equality $D(\mathtt{S}, \mathbf{k}') = D(\mathtt{S}, \mathbf{k})$ holds.

If $\mathbf{k}' \notin \mathtt{A}_{\mathrm{MM}}$, set $\mathbf{k} = \mathbf{k}'$, and find a new arrangement $\mathtt{D}^{\mathbf{k}'}$ that satisfies conditions (3.9)-(3.11). Because $|\mathtt{A}|$ is finite and each $\mathbf{k}'$ leads to a different member of $\mathtt{A}$ the sequence of arrangement indicators $\mathbf{k}, \mathbf{k}', \ldots, \mathbf{k}^{(\kappa)}$ must reach some $\mathbf{k}^{(\kappa)} \in \mathtt{A}_{\mathrm{MM}}$ such that $D\left(\mathtt{S}, \mathbf{k}^{(\kappa)}\right) = D(\mathtt{S}, \mathbf{k})$. By Corollary 2, $D\left(\mathtt{S}, \mathbf{k}^*\right) \leq D\left(\mathtt{S}, \mathbf{k}^{(\kappa)}\right)$, and so, by our initial assumption, $D(\mathtt{S}, \mathbf{k}) = D\left(\mathtt{S}, \mathbf{k}^*\right)$. ∎

Theorem 1.

Let $\underline{\mathtt{S}}$ and $\underline{\mathbf{k}} \in \mathtt{A}$ be an optimal solution of **FMSP**. Let $\mathbf{k}^* \in \mathtt{A}_{\mathrm{M}}$ and let $\mathtt{S}_f^* \in \underset{\mathtt{S}_f \in \mathtt{S}_{\mathrm{F}}}{\operatorname{argmin}} D\left(\mathtt{S}_f, \mathbf{k}^*\right)$. Then $D\left(\mathtt{S}_f^*, \mathbf{k}^*\right) = D(\underline{\mathtt{S}}, \underline{\mathbf{k}})$.

Proof:

By assumption, $D(\underline{\mathtt{S}}, \underline{\mathbf{k}}) \leq D\left(\mathtt{S}_f^*, \mathbf{k}^*\right)$. Consider the arrangement $\mathtt{D}^*$ and the resulting $D\left(\underline{\mathtt{S}}, \mathbf{k}^*\right) = \sum_{i=1}^{n} \bar{d}_i^*(\underline{\mathtt{S}})$. Lemma 4 implies that $D(\underline{\mathtt{S}}, \underline{\mathbf{k}}) \geq D\left(\underline{\mathtt{S}}, \mathbf{k}^*\right)$, and $D\left(\underline{\mathtt{S}}, \mathbf{k}^*\right) \geq D\left(\mathtt{S}_f^*, \mathbf{k}^*\right)$ follows from $\mathtt{S}_f^* \in \underset{\mathtt{S}_f \in \mathtt{S}_{\mathrm{F}}}{\operatorname{argmin}} D\left(\mathtt{S}_f, \mathbf{k}^*\right)$, so $D(\underline{\mathtt{S}}, \underline{\mathbf{k}}) = D\left(\mathtt{S}_f^*, \mathbf{k}^*\right)$. ∎

Note that the results of Theorem 1 are invariant to row permutations on $\mathtt{D}^*$. More accurately, let $\pi(\bullet)$ denote a specific permutation. If for every $\mathbf{d}_s^{k_s} \in \mathtt{D}^{\mathbf{k}}$, $\mathbf{d}_s^{k_s} = \pi\left(\mathbf{d}_s^*\right)$, then $D\left(\mathtt{S}_f^*, \mathbf{k}\right) = D\left(\mathtt{S}_f^*, \mathbf{k}^*\right)$.

We now obtain the conditions required to extend Theorem 1 to **CCNIM-sp**. The theorem states that solving **FMSP** on a set of ordered scenarios yields $\bar{\mathbf{d}}^*$, where $\Pr\left\{\bar{\mathbf{d}}^* \geq \mathbf{d}_s^*\right\} \geq p$, and $\mathbf{1}^T \bar{\mathbf{d}}^*$ is minimal. Obviously, if there is some ordering of the ships for which $\underline{\mathbf{v}} \leq \bar{\mathbf{d}}^* \leq \bar{\mathbf{v}}$, then we could set $\mathbf{v}^* = \bar{\mathbf{d}}^*$, and $\mathbf{v}^*$ would be an optimal solution to

CCNIM-sp. If that condition does not hold, we require some transformation from an optimal solution of **FMSP** to that of **CCNIM-sp**.

We consider the effect of the lower bounds first. It is easy to see that if $\underline{\mathbf{v}} \nleq \overline{\mathbf{d}}^*$, then setting $\mathbf{v} = \max\left\{\overline{\mathbf{d}}^*, \underline{\mathbf{v}}\right\}$ may not solve **CCNIM-sp** optimally. For example, consider the case where only one of the two scenarios (5,1) and (4,3) must be successfully covered, and $\underline{\mathbf{v}} = (3,3)$. Then $\overline{\mathbf{d}}^* = (5,1)$ leads to the allocation (5,3), but the optimal solution to **CCNIM-sp** is (4,3).

We transform the data pertaining to **FMSP** to obtain an optimal solution that satisfies the lower bounds. If D, $\boldsymbol{\varphi}$, and p define an instance of **FMSP**, then we label by **FMSP-lb** an instance of **FMSP** on the following data (marked by the prime symbol): $\mathrm{D}' = \mathrm{D} \bigcup \underline{\mathbf{v}}$, $\varphi'_{\underline{\mathbf{v}}} = 0.5$, $\varphi'_s = 0.5\varphi_s \ \forall s \in \mathrm{S}$, and $p' = 0.5(1+p)$. By Theorem 1, we arrange D' monotonically and obtain an optimal solution $\overline{\mathbf{d}}'^*$ and associated minimal p'-feasible subset S'^*_f. Because $\sum_{s \in \mathrm{S}} \varphi'_s < p'$, we must have $\underline{\mathbf{v}} \in \mathrm{S}'^*_f$, so $\underline{\mathbf{v}} \leq \overline{\mathbf{d}}'^*$. Because $\varphi'_s = 0.5\varphi_s \ \forall s \in \mathrm{S}$ and $p' - \varphi'_{\underline{\mathbf{v}}} = 0.5p$, we have $\sum_{s \in \mathrm{S}'^*_f} \varphi_s \geq p$ so $\mathrm{S}'^*_f - \{\underline{\mathbf{v}}\}$ is a p-feasible subset in **FMSP**. Therefore, we may set $\mathbf{v} = \overline{\mathbf{d}}'^*$ as an optimal solution of **CCNIM-sp** as long as we list the ships monotonically by lower bound, such that $\underline{v}_1 \geq, ..., \geq \underline{v}_n$, and $\overline{\mathbf{d}}'^* \leq \overline{\mathbf{v}}$. But, we already assume that every scenario is feasible, i.e. there is some permutation of the demands $Y\mathbf{d}_s$ that satisfies the inequality $\overline{\mathbf{v}} \geq Y\mathbf{d}_s$. By Corollary 1, if $\overline{\mathbf{v}}$ is also non-increasing, then $\overline{\mathbf{v}} \geq \mathbf{d}^*_s \ \forall s$.

We conclude that if an ordering of the ships exists where $\underline{\mathbf{v}}$ and $\overline{\mathbf{v}}$ can be non-increasing simultaneously, we can optimally solve **CCNIM-sp** as a simpler single-stage problem by ordering the demand vectors monotonically *a priori*. If such an ordering does not exist, then, by ordering the ships such that $\overline{\mathbf{v}}$ is non-increasing, we guarantee that the solution of **FMSP-lb** is a feasible solution of **CCNIM-sp**. It seems reasonable to assume that both bounds be non-increasing simultaneously, because the lower bounds are generally fixed for all ships, or are set proportionally to the capacities. Note that the resulting allocation plan may not be unique, as there may be alternative plans of equal cost that satisfy a different subset of scenarios. From this point forward, we assume that ships are listed monotonically by capacities, that the lower bounds vector is monotonic, and that all demand vectors are monotonically ordered unless otherwise specified.

Of course, the MAP constitutes a restriction of **CCNIM**, and may increase the cost of an optimal solution through an increased depot inventory. When calculating the procurement requirements by **CCNIM**, an unrestricted assignment plan may assign ships with low inventories to high-demand missions in period I in order to spare missiles for period II in scenarios we do not intend to satisfy. Furthermore, a different assignment plan alters the numbers of missiles remaining on individual ships in satisfied scenarios, and may reduce the number of missiles left on ships where they are not required in period II. For these two reasons, assignments that lead to a smaller depot requirement may be found if the MAP is not enforced.

Returning to the example that follows Lemma 1, consider the situation with three targets, corresponding demand vector $\mathbf{d} = (4,4,2)$, and an inventory vector for three shooters of $\mathbf{v} = (4,3,1)$. Under the MAP, no ship has any missiles left following the engagement, but the scenario is not satisfied. If the commander were to assign the target requiring two missiles to the ship carrying four missiles, then there would be two missiles left following the engagement. These missiles could reduce the depot requirement for period II. However, as mentioned in Chapter II, it appears that in many cases a monotonic plan is optimal for **CCNIM**.

C. CCNIM-MAP

Because enforcing the MAP is desirable from an operational perspective, and the cost of its enforcement appears to be small, we modify **CCNIM** to incorporate the MAP in “**CCNIM-MAP**”:

$$(\textbf{CCNIM-MAP}) \quad \min_{x^{\mathrm{I}}, x^{\mathrm{II}}, \mathbf{v}^{\mathrm{I}}} c_1 x^{\mathrm{I}} + c_2 x^{\mathrm{II}} \tag{3.12}$$

s.t.

$$\Pr\left\{\mathbf{v}^{\mathrm{I}} \geq \tilde{\mathbf{d}}^{\mathrm{I}}\right\} \geq p^{\mathrm{I}} \tag{3.13}$$

$$\mathbf{1}^T \mathbf{v}^{\mathrm{I}} - x^{\mathrm{I}} = 0 \tag{3.14}$$

$$\underline{\mathbf{v}} \leq \mathbf{v}^{\mathrm{I}} \leq \overline{\mathbf{v}} \tag{3.15}$$

$$\mathbf{v}^{\mathrm{I}} \in \mathbb{Z}_+^n \tag{3.16}$$

$$\text{where} \quad \mathbf{r} \equiv \left(\mathbf{v}^{\mathrm{I}} - \tilde{\mathbf{d}}^{\mathrm{I}}\right)^+ \tag{3.17}$$

$$\mathbf{v}^{\mathrm{II}} \equiv \mathbf{r} + \mathbf{w}\left(\mathbf{r}, F_{\mathrm{II}}\left(\bullet | \tilde{\mathbf{d}}^{\mathrm{I}}\right)\right) \tag{3.18}$$

and where $\mathbf{w}(\bullet) \in \mathbb{Z}_+^n$ is chosen such that

$$\Pr\left\{\mathbf{v}^{\mathrm{II}} \geq Y_{\mathrm{MAP}}^{\mathrm{II}}\left(\mathbf{v}^{\mathrm{II}}\right) \tilde{\mathbf{d}}^{\mathrm{II}}\right\} \geq p_s^{\mathrm{II}} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{3.19}$$

$$\mathbf{1}^T \mathbf{w}\left(\mathbf{r}, F_{\mathrm{II}}\left(\bullet | \tilde{\mathbf{d}}^{\mathrm{I}}\right)\right) - x^{\mathrm{II}} \leq 0 \tag{3.20}$$

$$\underline{\mathbf{v}} - \mathbf{r} \leq \mathbf{w}\left(\mathbf{r}, F_{\mathrm{II}}\left(\bullet | \tilde{\mathbf{d}}^{\mathrm{I}}\right)\right) \leq \overline{\mathbf{v}} - \mathbf{r} \tag{3.21}$$

$$\text{where} \quad Y_{\mathrm{MAP}}^{\mathrm{II}}(\mathbf{v}) \in \arg_{Y \in \Upsilon}\left\{\left(Y^{-1}\mathbf{v}\right)_i - \left(Y^{-1}\mathbf{v}\right)_{i+1} \geq 0 \quad \forall i \leq n-1\right\} \tag{3.22}$$

and with these restrictions on the data:

$$d_{is}^{\mathrm{I}} \geq d_{i+1,s}^{\mathrm{I}} \;\; \forall i \leq n-1, \forall s \in \mathbb{S}^{\mathrm{I}} \tag{3.23}$$

$$d_{is'}^{\mathrm{II}} \geq d_{i+1,s'}^{\mathrm{II}} \;\; \forall i \leq n-1, \forall s' \in \mathbb{S}^{\mathrm{II}}. \tag{3.24}$$

$$\overline{v}_i \geq \overline{v}_{i+1} \;\; \forall i \leq n-1 \tag{3.25}$$

CCNIM-MAP is only a two-stage model, and is, therefore, simpler than **CCNIM**. The flexibility in target assignment allows us to preprocess the demand vectors, and order each one monotonically. We assign demand d_{is}^{I} to ship i in every period-I scenario, and calculate the remaining inventory r_i on each ship. As in **CCNIM**, we then decide the number of missiles w_i^{II} to add to each ship's inventory for period II. This determines $\mathbf{v}^{\mathrm{II}}$, the period-II inventories. To enforce the MAP, some permutation $Y_{\mathrm{MAP}}^{\mathrm{II}}(\mathbf{v})$, which assigns larger demands to ships with larger inventories, must be determined. Because demands are ordered monotonically, this permutation is simply the inverse of the permutation, which arranges the ships inventories in non-increasing order. An inverse of a permutation matrix is well defined, because its determinant is either 1 or –1. The remainder of this dissertation is dedicated to finding practical solution methods to **CCNIM-MAP**.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. IDENTIFYING OPTIMAL SOLUTIONS

The IP model **RFFAM** may fail to reach an optimal solution in a reasonable length of time for problem instances that involve more than a few ships and/or scenarios, depending in the case specifications and the cost ratio used. Therefore, this chapter explores the properties of feasible solutions of **CCNIM-MAP**, and identifies several specific "points of interest," i.e., a small set of potentially optimal solutions that correspond to optimal solutions for certain cost ratios. These points may be found by solving instances of **RFFAM-sp** and restrictions of **RFFAM**. However, these methods prove unreliable, so in the next chapter we develop a specialized algorithm to identify these points based on the properties of PEPs.

A. STRUCTURE

The objective function (3.12) is the weighted sum of two variables, x^{I} and x^{II}, where x^{I} is the total number of missiles assigned to the ships in the period I, and x^{II} is the number of missiles stored at the depot. We define an integer pair $\mathbf{x} = \left(x^{\mathrm{I}}, x^{\mathrm{II}}\right)$ to be feasible if there exist missile allocations and target assignments that satisfy the probability requirements in **CCNIM-MAP**, with corresponding values of x^{I} and x^{II}. Because the objective function is entirely determined by the value of $\mathbf{x}$, we go as far as to refer to $\mathbf{x}$ as a *solution* to **CCNIM-MAP** in the following text (although knowing $\mathbf{x}$ does not reveal all of the other variables involved). All other variables are treated as secondary, and are needed only to assess the feasibility of $\mathbf{x}$. We refer to the value of the point $\mathbf{x}$ as $\mathbf{cx} = c_1 x^{\mathrm{I}} + c_2 x^{\mathrm{II}}$. In much of the following discussion, we refer to x_*^{I} as a first-period optimal solution and to the integer pair $\mathbf{x}_* = \left(x_*^{\mathrm{I}}, x_*^{\mathrm{II}}\right)$ as an optimal solution of the full problem.

We define the *quasi-feasible region* for **CCNIM-MAP**, $\mathtt{F} \subset \mathbb{Z}_+^2$, as the set of all feasible integer pairs $\left(x^{\mathrm{I}}, x^{\mathrm{II}}\right)$. We bound $\mathtt{F}$ by aggregating the individual ship-capacity constraints. Combining these with the lower bounds obtained for x^{I} and x^{II}, and the non-negativity restriction on x^{II}, we conclude that $\mathtt{F}$ is contained within the rectangle defined by $b^{\mathrm{I}} \le x^{\mathrm{I}} \le \mathbf{1}^T \overline{\mathbf{v}}$ and $0 \le x^{\mathrm{II}} \le \max_s b_s^{\mathrm{II}}$. Note that x^{II} is not explicitly bounded in **CCNIM-MAP**, but will not exceed $\max_s b_s^{\mathrm{II}}$, the greatest number of missiles we may need in period II, in any actual solution.

We also define the following subset of $\mathtt{F}$,

$$\mathtt{O}(\mathtt{F}) \equiv \left\{ \left(x^{\mathrm{I}}, x^{\mathrm{II}}\right) | \left(x^{\mathrm{I}}, x^{\mathrm{II}}\right) \in \mathtt{F},\ \left(x^{\mathrm{I}} - 1, x^{\mathrm{II}}\right) \notin \mathtt{F},\ \left(x^{\mathrm{I}}, x^{\mathrm{II}} - 1\right) \notin \mathtt{F} \right\}. \tag{4.1}$$

Only the points in $\mathtt{O}(\mathtt{F})$ are potentially optimal, because both points $\left(x^{\mathrm{I}} - 1, x^{\mathrm{II}}\right)$ and $\left(x^{\mathrm{I}}, x^{\mathrm{II}} - 1\right)$ have a lower objective value (assuming positive costs) than $\left(x^{\mathrm{I}}, x^{\mathrm{II}}\right)$. If either of those solutions is feasible, then it is obviously preferable to $\left(x^{\mathrm{I}}, x^{\mathrm{II}}\right)$. In other words, any optimal solution of **CCNIM-MAP** must be Pareto optimal [Rardin 1998, p. 379] with respect to the two variables x^{I} and x^{II}. To solve **CCNIM-MAP**, we only need to identify which $\mathbf{x} \in \mathtt{O}(\mathtt{F})$ is optimal, and in many cases we can do so without solving **RFFAM**.

We explore the properties of $\mathtt{O}(\mathtt{F})$ to find features that will lead us to specialized algorithms. KPP use a specialized decomposition algorithm to solve **CCGIM**, so we begin our exploration by examining the structure of $\mathtt{O}(\mathtt{F})$ for that model, which is simpler than **CCNIM-MAP**. In **CCGIM**, safety stocks exist that must be replenished, if used, and units may transship between themselves at the end of period I. Because of the safety stocks, for any given period-I scenario, all allocation plans expend the same amount of ordnance. Furthermore, KPP prove that every two-period optimal solution requires the same amount of ordnance in total. It is clear that $\mathtt{O}(\mathtt{F})$, in the KPP setting, is a line segment in $\mathbb{Z}_+^2$, maintaining the relation $x^{\mathrm{I}} + x^{\mathrm{II}} = const$.

The geometric view provides further intuition into the correctness of the KPP decomposition procedure. KPP seek the point $\mathbf{x}_a \equiv \underset{\left(x^{\mathrm{I}}, x^{\mathrm{II}}\right) \in \mathtt{O}(\mathtt{F})}{\operatorname{argmin}} x^{\mathrm{I}}$, which is the potentially optimal point that minimizes the number of missiles in the initial (period I) ship allocation. They obtain it by decomposing the two-period model into two separate one-period problems, which they solve sequentially. In the first problem, they minimize x_a^{I}, the amount of ordnance allocated to the units in order to satisfy demands in the first combat period with a specified probability. The specific period-I allocations are used as parameters in the period-II problems, where KPP minimize the amount of ordnance to be added to the units' inventories from the depot $\mathbf{1}^T\mathbf{w}_s$ following each possible period-I scenario s. By following this procedure, they obtain $x_a^{\mathrm{II}} = \max_s \{\mathbf{1}^T\mathbf{w}_s\}$. By the assumption that $c_1 \geq c_2$, $\mathbf{x}_a$ must be globally optimal for **CCGIM**.

The structure of $\mathtt{O}(\mathtt{F})$ may be more complex in the case of **CCNIM-MAP**. We define several specific integer pairs, and show which of these are optimal under which circumstances. As above, let $\mathbf{x}_a \equiv \underset{(x^{\mathrm{I}},x^{\mathrm{II}})\in\mathtt{O}(\mathtt{F})}{\operatorname{argmin}}\, x^{\mathrm{I}}$ be the potentially optimal point that minimizes the number of missiles in the initial (period I) ship allocation, and let $\mathbf{x}_d \equiv \underset{(x^{\mathrm{I}},x^{\mathrm{II}})\in\mathtt{O}(\mathtt{F})}{\operatorname{argmin}}\, x^{\mathrm{II}}$ be the potentially optimal point that minimizes the number of missiles stored in the depot. Let $\mathtt{X}(\mathtt{F}) \equiv \underset{(x^{\mathrm{I}},x^{\mathrm{II}})\in\mathtt{O}(\mathtt{F})}{\operatorname{argmin}}\, x^{\mathrm{I}} + x^{\mathrm{II}}$ be the set of points that minimize the total number of missiles.

By these definitions, it is obvious that in cases where there is a clear preference for allocation strategy, that is, when the ratio c_2/c_1 is very different from 1, we seek either the point $\mathbf{x}_a$ or the point $\mathbf{x}_d$ as the optimal solution. (It is difficult to determine how large or small "different from 1" must be without solving **CCNIM-MAP**.) When $c_2/c_1 = 1$, any point $\mathbf{x} \in \mathtt{X}(\mathtt{F})$ is an optimal solution. When the cost ratio is set such that $c_2/c_1 \approx 1$, then identifying the optimal points in $\mathtt{O}(\mathtt{F})$ may be more difficult.

The first point of note is that $\mathbf{x}_a$ may not belong to $\mathtt{X}(\mathtt{F})$, in which case it is not optimal when $c_1 = c_2$. As an example, consider case 4a in table 2, which has five combat ships, $\mathtt{S}^{\mathrm{I}} = \{s_1, s_2, s_3\}$ and $\mathtt{S}^{\mathrm{II}} = \{s_4'\}$. In this case, $\mathbf{x}_a = (7,2)$ because seven missiles are enough to satisfy scenario s_1 and meet the probability requirement. If that scenario actually occurs, we will have no inventory carried over into period II, so two missiles are needed in the depot. Increasing the period-I allocation total to eight allows the period-I allocation $\mathbf{v}^{\mathrm{I}} = (2,2,2,1,1)$, which covers scenarios s_2 and s_3. If either of these scenarios occurs, we obtain the (reordered) remainder vectors (1,1,0,0,0) or (1,1,1,0,0), respectively, both of which satisfy the period-II requirement without requiring any additional missiles. If scenario s_1 occurs, the remainder is (2,2,1,1,0). Hence we have the optimal solution $(8,0)$, which requires one missile less, in total, than $\mathbf{x}_a$.

The reason that $\mathbf{x}_a$ may not belong to $\mathtt{X}(\mathtt{F})$ is that a ship carries no safety stock. Therefore, the number of missiles expended in unsatisfied scenarios depends on the allocation. Furthermore, adding missiles to the ships' inventories in period I relaxes the limitations on the scenarios we may choose to satisfy in that period. We may now choose a different allocation plan, requiring more than x_a^{I} missiles, that uses fewer missiles in the unsuccessful period-I scenarios, thereby saving more missiles for period-II and lowering the requirement for depot stockpile. The difference between the depot requirements may exceed the difference between the total period-I allocations. We refer to this effect as the *no-safety-stock effect* in future discussion.

The reader may correctly assume that an enumeration procedure, which generates every p^{I}-feasible subset, can overcome these difficulties. Such an algorithm, however, is inefficient compared with the algorithm we provide in the next chapter, and solves **CCNIM-MAP** only for cost ratios where $c_2 / c_1 \leq 1$. We detail such an algorithm in Appendix D.

φ_s	(0.5, 0.25, 0.25)
$\varphi_{s'\|s}$	$1\ \forall s \in S^{\mathrm{I}}$
p^{I}	0.5
p_s^{II}	1
d_{ms} $s \in S^{\mathrm{I}}$ $s' \in S^{\mathrm{II}}$	$\begin{array}{ccccc} & s_1 & s_2 & s_3 & s'_4 \\ m_1 & 7 & 2 & 1 & 1 \\ m_2 & 0 & 2 & 1 & 1 \\ m_3 & 0 & 2 & 1 & 0 \\ m_4 & 0 & 0 & 1 & 0 \\ m_5 & 0 & 0 & 1 & 0 \end{array}$
$\underline{v}_i$	$0\ \forall i$
$\overline{v}_i$	$8\ \forall i$

Table 2. Case 4a - Example Where $\mathbf{x}_a \notin \mathrm{X}(\mathrm{F})$.

The table lists the probabilities of each scenario (φ_s, $\varphi_{s'|s}$), the success thresholds specified for each period (p^{I}, p_s^{II}), the demands associated with the various missions in each scenario d_{ms}, and the capacity limits imposed on each ship ($\underline{v}_i$, $\overline{v}_i$). A minimum period-I allocation leads to the solution $\mathbf{x}_a = (7,2)$. If we allow 8 missiles to be allocated to the ships in the first period, we can choose an allocation that satisfies the scenarios s_2 and s_3, instead of scenario s_1. The remaining inventory following any of the three scenarios will satisfy scenario s'_4.

The no-safety-stock effect causes a second difficulty in the analysis of $\mathrm{O}(\mathrm{F})$. **CCNIM-MAP** assumes that ships will expend their entire inventory when the assigned demand exceeds that inventory. Increasing the allocation to any such ship will increase the number of missiles it expends in such a scenario. However, if the original allocation plan was already p^{I}-feasible, then increasing it constitutes a “waste” of missiles (from the logistics point of view), because those missiles will not necessarily reduce the depot requirements for period II. Thus we may have cases where $\left(x^{\mathrm{I}}, x^{\mathrm{II}}\right) \in \mathtt{F}$, but $\left(x^{\mathrm{I}}+1, x^{\mathrm{II}}-1\right) \notin \mathtt{F}$.

The point $\mathbf{x}_d$ may also not belong to the set $\mathtt{X}(\mathtt{F})$, due to the no-safety-stock effect, as well as the *no-transshipment effect*. The no-transshipment effect arises because, following period I, the number of missiles remaining on some ships may exceed the number of missiles actually required by them to satisfy period-II scenarios. By assumption, these extra missiles are essentially wasted because they cannot be transshipped to other ships that may require them.

We illustrate these effects by examining $\mathtt{O}(\mathtt{F})$ for case 4b in Table 3. There, $\mathbf{x}_a = (15{,}15) \in \mathtt{X}(\mathtt{F})$, and the initial allocation of (5,4,4,2) satisfies every scenario but s_4, whose associated demand vector is (5,4,4,3) when ordered monotonically. We also know that $(18{,}12) \in \mathtt{X}(\mathtt{F})$, and the reader may verify that (19,11) is not. That is because there is no p^{I}-feasible allocation of 19 missiles that does not expend at least 16 missiles in any monotonic assignment plan if scenario s_4 occurs. In fact, there are exactly five monotonic 19-missile allocations that would expend only 16 missiles, e.g., (7,4,4,4). There are three total missiles remaining following s_4, just as when we optimally allocate 18 missiles. Hence, the depot requirement, which is driven by s_4, is still 12, and the point (19,12) is feasible. Note that $(19{,}12) \notin \mathtt{O}(\mathtt{F})$ because $(18{,}12) \in \mathtt{O}(\mathtt{F})$, so the next point (in order of decreasing x^{II}) that belongs to $\mathtt{O}(\mathtt{F})$ is (20,11), and the sum of missiles expended has increased by one. The reader may verify that a similar increase in total expenditure occurs from the point $(26,5) \in \mathtt{O}(\mathtt{F})$ to $(28,4) \in \mathtt{O}(\mathtt{F})$, because 27 missiles cannot be allocated in period I without expending 17 missiles when scenario s_4 occurs. Thus, we require the same depot level as when we allocate 26 missiles optimally, and $(27,4) \notin \mathtt{O}(\mathtt{F})$.

φ_s	$\left(\frac{2}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6}\right)$
$\varphi_{s'\|s}$	$(0.4,0.2,0.2,0.2)\ \forall s\in\{s_1,\dots,s_5\}$
p^{I}	0.83
p_s^{II}	$0.8\ \forall s$
d_{ms} $s\in\mathbb{S}^{\mathrm{I}}$	$\begin{array}{cccccc} & s_1 & s_2 & s_3 & s_4 & s_5 \\ m_1 & 3 & 4 & 5 & 5 & 5 \\ m_2 & 3 & 3 & 4 & 5 & 4 \\ m_3 & 3 & 2 & 4 & 4 & 4 \\ m_4 & 0 & 1 & 2 & 3 & 1 \end{array}$
d_{ms} $s'\in\mathbb{S}^{\mathrm{II}}$	$\begin{array}{ccccc} & s'_6 & s'_7 & s'_8 & s'_9 \\ m_1 & 3 & 4 & 5 & 5 \\ m_2 & 3 & 3 & 4 & 5 \\ m_3 & 3 & 2 & 4 & 4 \\ m_4 & 0 & 1 & 2 & 3 \end{array}$
$\underline{v}_i$	$2\ \forall i$
$\overline{v}_i$	$8\ \forall i$

Table 3. Parameter Specifications for Case 4b.
All labels defined as in Table 2.

The increases in the number of missiles expended, shown above, result from the no-safety-stock effect. Had there been some safety stock, which must be replenished, the ships would have expended 17 missiles whenever scenario s_4 occurred, regardless of the initial allocation. It is perhaps counter-intuitive that optimal allocations do <u>not</u> satisfy every demand, even when enough missiles are allocated in total for that period.

When minimizing x^{II}, it happens that $(32,0)\notin\mathbb{O}(\mathbb{F})$ and $\mathbf{x}_d=(32,1)$. This further increase in total missile requirement results from the no-transshipment effect, and occurs when x^{I} increases from 29 missiles to 30, requiring a depot inventory of 3 missiles in both cases. The period-II scenario specifications are independent of the period-I scenarios, and $b_s^{\text{II}} = 15$ for all s, corresponding to the period-II minimal allocation of (5,4,4,2). There are only two (monotonic) period-I allocations using 30 missiles: (8,8,8,6) and (8,8,7,7). If scenario s_4 occurs, the remainders are (3,3,4,3) and (3,3,3,4), respectively, which are essentially equivalent. Three more missiles are required, in total, to increase the inventories of three ships to the requirement of the period-II minimal allocation, but the fourth ship, which requires an inventory level of two missiles, is also carrying three and has one in excess. 29 missiles can be initially allocated according to (8,8,8,5), leaving (3,3,4,2) missiles if scenario s_4 occurs, and still requiring a depot inventory of three missiles.

Figure 2 gives a full mapping of $\mathtt{O}(\mathtt{F})$ for case 4b of **CCNIM-MAP**, based on the discussion above. Based on the above analysis, we find all of the optimal solutions for any cost ratio, and summarize the results in Table 4. Some solutions cover a span of ratios, while others are optimal for only a specific cost ratio; these latter solutions are never uniquely optimal. The solutions obtained by solving different instances of case 4b in **RFFAM,** differentiated by cost-ratios, correspond to some of the points of interest in $\mathtt{O}(\mathtt{F})$, as can be seen in Table 4.

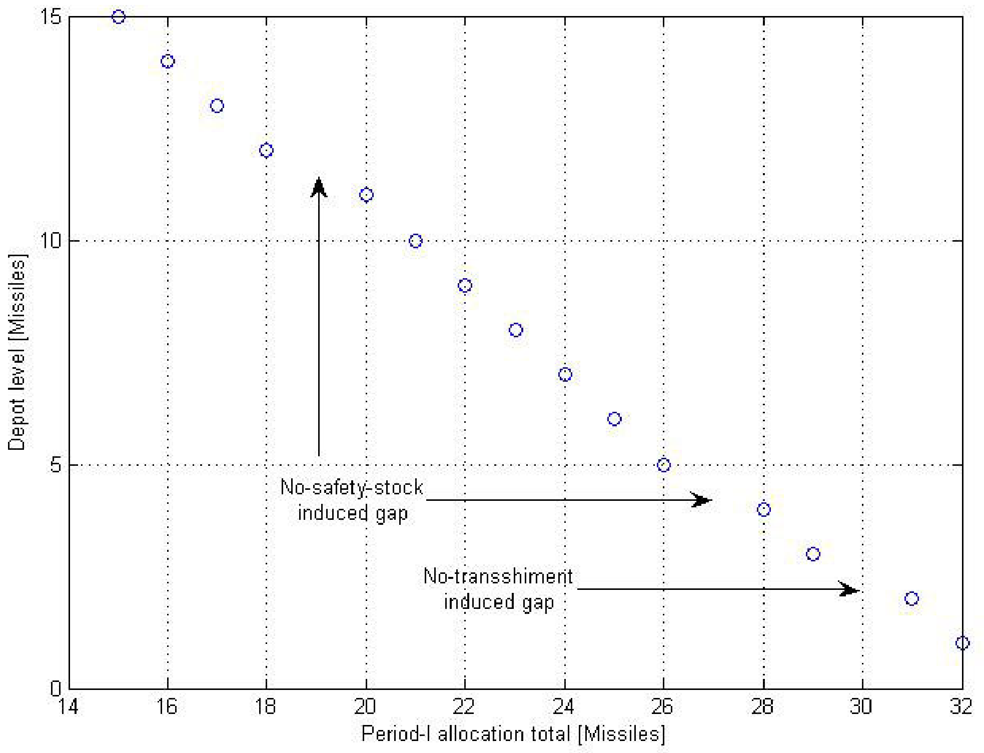

Figure 2. A Map of $O(F)$ for Case 4b in Table 3.

The figure shows every point belonging to $O(F)$ when solving case 4b. Only these points could lead to optimal solutions of **CCNIM-MAP**. There are three occasions where $O(F)$ breaks from linearity. Two breaks occur due to the no-safety-stock effect, which results in no feasible solution to **CCNIM-MAP** if we use the total period-I allocation and depot inventory pairs (19,11) or (27,4). The pair (30,2) also leads to no feasible solution of **CCNIM-MAP**, due to the no-transshipment effect.

c_2/c_1	Optimal Solutions, $\mathbf{x}_* = \left(x_*^{\mathrm{I}}, x_*^{\mathrm{II}}\right)$	**RFFAM** Initial Allocation
<1	(15,15)	(5,4,4,2)
1.0	(15,15), (16,14), (17,13), (18,12)	(8,4,4,2)
$1<\cdots<8/7$	(18,12)	(8,4,4,2)
$8/7$	(18,12), (26,5)	(8,4,4,2)
$8/7<\cdots<3/2$	(26,5)	(8,8,8,2)
$3/2$	(26,5), (29,3), (32,1)	(8,8,8,8)
$>3/2$	(32,1)	(8,8,8,8)

Table 4. Optimal Solutions for Case 4b in Table 3.

The "c_2/c_1" column gives the cost ratio, thereby completing the definition of the instance. The "Optimal Solution" column gives the optimal number of missiles to be loaded for period-I combat and the number of missiles to be stored for later use. "**RFFAM** Initial Allocation" gives initial load-outs for the ships, as obtained by solving **RFFAM** for the corresponding instance of case 4b. Notice there are several specific ratios that have multiple optimal solutions.

Incidentally, if the cost ratio is greater than 1.5, the optimal solution (32,1) is a direct result of the MAP, and is of strictly higher cost than the optimal solution in the unrestricted case. Consider the initial allocation (8,8,8,7), requiring a total of 31 missiles. If the targets of scenario s_4 are assigned according to (5,4,3,5), then the remainders are (3,4,5,2), and only a single extra missile must be stored at the depot. For any other period-I scenario, no more missiles would be required. Hence, the solution (31,1) is feasible, and costs less than $\mathbf{x}_d = (32,1)$.

Finally, note that loading the ships to full capacity guarantees that the resulting depot inventory is minimal. However, it may be that $\left(\mathbf{1}^T\bar{\mathbf{v}} - k, x_d^{\mathrm{II}}\right) \in \mathcal{F}$ for some integer $k > 0$, so the point $\left(\mathbf{1}^T\bar{\mathbf{v}}, x_d^{\mathrm{II}}\right)$ may not belong to $\mathcal{O}(\mathcal{F})$. This can be caused by either the no-transshipment effect or by the no-safety-stock effect because both effects cause portions of the convex hull of $\mathtt{F}$ to parallel the x^{I} axis.

B. EXAMPLE OF RFFAM-BASED TECHNIQUES

As observed in Chapter II, case 2f is difficult to solve by **RFFAM**. In this section, we attempt to solve it for various cost ratios by exploiting our newly gained insight into the feasible region and by exploiting other techniques involving variations of **RFFAM**.

The points $\mathbf{x}_a$ and $\mathbf{x}_d$ correspond to "preemptive" allocation policies that respectively minimize the total number of missiles allocated to the ships in period I, or minimize the number of missiles that are kept in the depot. We can try to use **RFFAM** to identify $\mathbf{x}_a$ and $\mathbf{x}_d$ directly by setting extreme cost ratios. However, even setting a cost ratio of 100 to 1, in either direction, does not yield an integer solution in one hour of computation.

We may be able to identify $\mathbf{x}_a$ and $\mathbf{x}_d$ significantly faster by taking advantage of their preemptive nature. The point $\mathbf{x}_a$ corresponds to a solution that minimizes the number of missiles allocated to ships in period I, and then minimizes the number of missiles held in inventory for that allocation. We know that $x_a^{\mathrm{I}} = b^{\mathrm{I}}$, where b^{I}, the minimum feasible x^{I}, is obtained by solving **RFFAM-sp**. Let $x^{\mathrm{II}}\left(x^{\mathrm{I}}\right)$ denote the value of x^{II} in an optimal solution of **RFFAM** where x^{I} is fixed. Because x^{II} is unbounded, a solution $x^{\mathrm{II}}\left(b^{\mathrm{I}}\right)$ must exist. We expect that solving **RFFAM** with the added constraint $\mathbf{1}^T\mathbf{v} = b^{\mathrm{I}}$ will yield the point $\mathbf{x}_{\mathrm{a}} = \left(b^{\mathrm{I}}, x^{\mathrm{II}}\left(b^{\mathrm{I}}\right)\right)$ faster than by setting $c_2 / c_1 \ll 1$ in **RFFAM**. However, for case 2f, an integer solution is not obtained in one hour even when **RFFAM** is restricted in that manner.

If we prefer to reduce the operational burden of replenishment operations, we seek the solution $\mathbf{x}_d$, which first minimizes x^{II} and then minimizes x^{I}. To do so, we can solve **RFFAM** twice. First, we fix an assignment plan that sets x^{I} to its upper bound $\overline{x}^{\mathrm{I}}$

by fixing $v_i = \overline{v}_i \; \forall i$, so that each ship is loaded to capacity. Solving **RFFAM**, we obtain $x_d^{\text{II}} = x^{\text{II}}\left(\overline{x}^{\text{I}}\right)$, the minimum depot inventory. We now fix $x^{\text{II}} = x_d^{\text{II}}$ and solve **RFFAM** to obtain $\mathbf{x}_d = \left(x^{\text{I}}\left(x_d^{\text{II}}\right), x_d^{\text{II}}\right)$, where $x^{\text{I}}\left(x^{\text{II}}\right)$ denotes the value of x^{I} in the solution of **RFFAM** for a fixed value of x^{II}. Obtaining a solution by this method should be significantly faster than simply solving **RFFAM** with $c_2/c_1 \gg 1$ because the number of potential allocations is reduced. When we fix the allocations of the 8 ships in case 2f to their upper bounds, we obtain the solution $x^{\text{II}} = 19$ in 5.3 seconds of computation. However, when we solve **RFFAM** again, with the depot inventory fixed, no integer solution is found in one hour.

Although the restrictions designed to yield $\mathbf{x}_a$ and $\mathbf{x}_d$ do not improve the performance of **RFFAM** significantly, we find other restrictions that do. In particular, one restriction, denoted **RFFAM-mII** ("mII" stands for monotonic in period II), solves case 2f in less that one second for every cost ratio. In **RFFAM-mII**, we restrict the period-II assignment variables by adding the constraints

$$u_{ims's}^{\text{II}} = 0 \quad \forall i,\ m \neq i,\ s \in \mathbb{S}^{\text{I}},\ s' \in \mathbb{S}^{\text{II}}(s)\,. \tag{4.2}$$

Recall that $u_{ims's}^{\text{II}}$ is set to 1 if ship i is assigned mission m in period-II scenario s', following scenario s in period I, and 0 otherwise. Because the targets have been monotonically ordered by demands, ship i is now assigned the ith largest demand in each period. This restriction may not be optimal because it ignores the numbers of missiles remaining on each ship following period I.

Lower bounds on the minimum number of missiles that must be procured to satisfy **RFFAM** can be found by relaxing the integrality constraints on some of the variables in **RFFAM**. We suggest removing the binary requirement from the variables controlling period-II assignments. An initial attempt to solve case 2f with $u_{ims's}^{\text{II}} \in [0,1]$ yields no optimal solutions in one hour of computation, but when $u_{ims's}^{\text{II}}$ is unrestricted, an optimal solution is obtained in about four seconds for every cost ratio. We refer to this model as **RFFAM-rx** ("rx" stands for relaxation). **RFFAM-rx** solves significantly faster than **RFFAM** itself, provides an integer allocation plan, and the optimal relaxed objective value provides a lower bound on the number of required missiles.

For cost ratios that satisfy $c_2 < c_1$, **RFFAM-rx** and **RFFAM-mII** yield the same initial allocation plan (8,7,7,6,5,4,3,2) and depot inventory (40), which can therefore be declared optimal. When $c_2 = c_1$, the plans are not identical, but are of equal cost. Therefore, we can declare the solution given by **RFFAM-mII** optimal. When $c_2/c_1 \geq 2$, however, the solutions of the two models do not have the same cost. **RFFAM-rx** yields a period-I allocation with all ships loaded to their capacities (64 missiles in all), and 19 missiles are required at the depot. The initial allocation prescribed by **RFFAM-mII** is $\mathbf{v}^{\mathrm{I}} = (8,8,8,8,8,8,6,3)$, with 26 missiles required at the depot. Although both solutions require the same number of missiles, the optimality gap is about 7%. We already know that $x_d^{\mathrm{II}} = 19$, so the solution obtained from **RFFAM-rx** is feasible, but in general that may not be the case.

It is possible that investing further effort can lead to dependable and fast solutions of **CCNIM-MAP** based on **RFFAM**, for problems with eight ships and eight scenarios in each period. But we require optimal solutions for significantly larger cases: this prompts the development of a specialized algorithm in the next chapter. The specialized algorithm can identify $\mathbf{x}_a$, $\mathbf{x}_d$, or some $\mathbf{x} \in \mathtt{X}(\mathtt{F})$, depending on the cost ratio, without requiring an LP-based solution at all.

THIS PAGE INTENTIONALLY LEFT BLANK

V. SPECIALIZED ALGORITHMS FOR CCNIM-MAP

This chapter develops a specialized enumerative algorithm that calculates the values of the points discussed in the previous chapter, and therefore solves **CCNIM-MAP** for a wide range of cost ratios. The specialized algorithm calculates the value of either $\mathbf{x}_a$ or $\mathbf{x}_d$ if $c_1 \neq c_2$, and at least one point from $\mathtt{X}(\mathtt{F})$ if $c_1 = c_2$. It does so by decomposing the two-period problem into single-period problems and using the remainders following period I as data for the period-II problem. However, as discussed in the previous chapter, the optimal period-I allocation and the resulting optimal depot inventory need not be an optimal solution of **CCNIM**. Instead, the algorithm we propose enumerates a set of candidate period-I allocations, calculates the depot inventory requirement resulting from each, and identifies the minimum-cost solution from these candidates.

A. DECOMPOSITION

The period-I problem comprises constraints (3.13), (3.15), and (3.16) from **CCNIM-MAP**. To simplify constraints (3.15), we replace the demand vector in (3.13) with the scenario *requirements* vectors $\widetilde{\mathbf{dd}}^{\mathrm{I}} = \max\left\{\tilde{\mathbf{d}}^{\mathrm{I}}, \underline{\mathbf{v}}\right\}$, which represent the minimum allocations required in satisfied scenarios. We obtain the following formulation of the period-I problem, **CCNIM-pI**.

(**CCNIM-pI**)
$$\min_{\mathbf{v}^{\mathrm{I}}} c_1 \mathbf{1}^T \mathbf{v}^{\mathrm{I}} \tag{5.1}$$

s.t.

$$\Pr\left\{\mathbf{v}^{\mathrm{I}} \geq \widetilde{\mathbf{dd}}^{\mathrm{I}}\right\} \geq p^{\mathrm{I}} \tag{5.2}$$

$$\mathbf{v}^{\mathrm{I}} \leq \overline{\mathbf{v}} \tag{5.3}$$

$$\mathbf{v}^{\mathrm{I}} \in \mathbb{Z}_+^n \tag{5.4}$$

As discussed in Chapter IV, an optimal solution of **CCNIM-MAP** must specify a period-I allocation that is feasible in **CCNIM-pI**. **CCNIM-pI**'s feasible region is characterized by the PEPs of the distribution of $\widetilde{\mathbf{dd}}^{\mathrm{I}}$, $\mathbf{v}_j^{\mathrm{I}}$, as seen by reformulating constraint (5.2) as $\mathbf{v}^{\mathrm{I}} \in \bigcup_{j\in J}\left(\mathbf{v}_j^{\mathrm{I}} + \mathbb{Z}_+^n\right)$ (see Chapter I for details).

For any given period-I allocation $\mathbf{v}_j^{\mathrm{I}}$, we derive $\left|S^{\mathrm{I}}\right|$ period-II problems, each one being a PIP. In each, we calculate x_{sj}^{II}, the depot requirement for each combination of period-I allocation and scenario. Because we require success in period II for every period-I scenario, the depot inventory requirement induced by each allocation plan is $x^{\mathrm{II}}\left(\mathbf{v}_j^{\mathrm{I}}\right) = \max_s x_{sj}^{\mathrm{II}}$. The remainders vector, $\hat{\mathbf{r}}_{sj} \equiv \left(\mathbf{v}_j^{\mathrm{I}} - \mathbf{d}_s^{\mathrm{I}}\right)^+$, is used as data in this model, which we call **CCNIM-pII**. ("pII" stand for "period II").

$$(\textbf{CCNIM-pII}) \qquad \min_{\mathbf{w}_{sj}^{\mathrm{II}},\, x_{sj}^{\mathrm{II}}} x_{sj}^{\mathrm{II}} \tag{5.5}$$

s. t.

$$\Pr\left\{\mathbf{v}_{sj}^{\mathrm{II}} \geq \left(Y_{\mathrm{MAP}}^{\mathrm{II}}\left(\mathbf{v}_{sj}^{\mathrm{II}}\right)\right)\tilde{\mathbf{d}}^{\mathrm{II}}\right\} \geq p_s^{\mathrm{II}} \tag{5.6}$$

$$\mathbf{1}^T \mathbf{w}_{sj} - x_{sj}^{\mathrm{II}} = 0 \tag{5.7}$$

$$\underline{\mathbf{v}} - \hat{\mathbf{r}}_{sj} \leq \mathbf{w}_{sj} \leq \overline{\mathbf{v}} - \hat{\mathbf{r}}_{sj} \tag{5.8}$$

$$\mathbf{w}_{sj} \in \mathbb{Z}_+^n \tag{5.9}$$

$$\text{where} \quad \mathbf{v}_{sj}^{\mathrm{II}} \equiv \hat{\mathbf{r}}_{sj} + \mathbf{w}_{sj} \tag{5.10}$$

$$\text{and } Y_{\mathrm{MAP}}^{\mathrm{II}}\left(\mathbf{v}\right) \in \arg_{Y\in\Upsilon}\left\{\left(Y^{-1}\mathbf{v}\right)_i - \left(Y^{-1}\mathbf{v}\right)_{i+1} \geq 0 \quad \forall i \leq n-1\right\} \tag{5.11}$$

Because the vectors $\hat{\mathbf{r}}_{sj}$ need not be monotonic, we may wish to allocate missiles to the ships in some order other than lexicographically, and assign targets accordingly. However, because of the MAP, the demand in the ith component of $\tilde{\mathbf{d}}^{\mathrm{II}}$ is always assigned to the same ship. Therefore, we may determine the mission assignments simultaneously with the allocations themselves. The assignments are determined by any permutation matrix that orders the chosen allocations monotonically to match large demands with large inventories. In the formulation of **CCNIM-pII**, we apply the inverse permutation, denoted $Y_{\mathrm{MAP}}^{\mathrm{II}}\left(\mathbf{v}\right)$, to the demand vector. For example, consider the very simple case where the two demand vectors are (4,4) and (6,3), and only one must be satisfied. If the remaining inventory is (3,6), we can satisfy the second scenario without allocating any more missiles to the ships.

As in the period-I problem, the period-II missile requirements take the inventory lower bounds into account. Because ships will not return to port if they do not need to load more missiles to satisfy their projected demands in the scenarios we plan to satisfy, the number of missiles required by each ship from the depot is the positive part of the difference between the demands (adjusted for lower bounds) and the remainders. If the difference is negative, then the excess missiles are "wasted" due to the no-transshipment policy (although they may be used by the ship in scenarios we are not planning to satisfy). Hence, the number of missiles each ship requires from the depot to satisfy period-II scenario s', following period-I scenario s, is given by $\mathbf{dd}^{\text{II}}_{s'|s,j} = \left(\max\left\{Y^{\text{II}}_{\text{LL}}\left(\mathbf{v}^{\text{II}}_{sj}\right)\mathbf{d}^{\text{II}}_{s'|s}, \underline{\mathbf{v}}\right\} - \mathbf{r}_{sj}\right)^{+}$, if period-I allocation j is chosen. We let $\widetilde{\mathbf{dd}}^{\text{II}}_{sj}$ denote the *n*-dimensional, integer, random vector of demands. Obviously, to determine the requirement prior to solving **CCNIM-pII**, we must be able to determine the optimum permutation matrix $Y^{\text{II}}_{\text{MAP}}\left(\mathbf{v}^{\text{II}}_{sj}\right)$ before choosing $\mathbf{v}^{\text{II}}_{sj}$.

As a first step in solving **CCNIM-pII**, we claim that if ship capacities are equal, constraint (5.8) can be dropped, and the optimal permutation of demands lists them in the same order as the remainders. For notational convenience, we may view this as a permutation of the remainders in non-increasing order. If ships have unequal capacities, a more elaborate procedure would be required to ensure that the specified allocations can actually be loaded onto the ships, but we ignore this complication in this dissertation, and assume that $\overline{v}_i = \overline{v}\ \forall i$. The validity of the sorting operation on the remainders is provided by the following Lemma 5, below.

Let $\mathbf{r} \in \mathbb{Z}^n_+$ represent the remainders following an arbitrary period-I scenario. There are $n!$ permutations of the elements of $\mathbf{r}$. Let $\mathrm{K} = \{1,...,n!\}$ be the index set of all permutations, and let $\mathbf{r}^k$, $k \in \mathrm{K}$, denote the kth permutation of $\mathbf{r}$. We denote the monotonically ordered remainders vector by $\mathbf{r}^*$, and let k^* denote some corresponding permutation index.

Lemma 5.

Let $\mathrm{D} = \bigcup_s \{\mathbf{d}_s\}$ be a set of scenarios, with corresponding index set S, where each column s is ordered in non-increasing order. Let $\mathbf{r}^k, k \in \mathrm{K}$ be a permutation of the remainders, and let k^* be a permutation index such that the components of $\mathbf{r}^*$ are non-increasing. Letting $x_{sk}^{\mathrm{II}} = \sum_i \left(d_{is} - r_i^k\right)^+$ be the required depot inventory if scenario s occurs, for remainders permutation $\mathbf{r}^k$, we have $x_{sk^*}^{\mathrm{II}} = \min_k x_{sk}^{\mathrm{II}}$.

Proof:

Let $k \in \operatorname{argmin}_{k \in \mathrm{K}} x_{sk}^{\mathrm{II}}$, and suppose that $\mathbf{r}^k \neq \mathbf{r}^*$. Then, there exist indices l and m such that $l < m$ and $r_l^k < r_m^k$. Consider $\mathbf{r}^{k'}$, an alternative permutation of $\mathbf{r}$, in which

$$r_m^{k'} = r_l^k \tag{5.12}$$

$$r_l^{k'} = r_m^k \tag{5.13}$$

$$r_i^{k'} = r_i^k \quad \forall i \neq l, m \tag{5.14}$$

We obtain

$$x_{sk'}^{\mathrm{II}} - x_{sk}^{\mathrm{II}} = \left[\left(d_{ls} - r_m^k\right)^+ + \left(d_{ms} - r_l^k\right)^+\right] - \left[\left(d_{ls} - r_l^k\right)^+ + \left(d_{ms} - r_m^k\right)^+\right]. \tag{5.15}$$

Because we assume $d_{ls} \geq d_{ms}$ and $r_l^k < r_m^k$, there are six cases to examine:

- If $d_{ls} \leq r_l^k$, then equation (5.15) reduces to $[0+0]-[0+0]=0$.
- If $d_{ms} \geq r_m^k$, we have $\left[\left(d_{ls} - r_m^k\right) + \left(d_{ms} - r_l^k\right)\right] - \left[\left(d_{ls} - r_l^k\right) + \left(d_{ms} - r_m^k\right)\right] = 0$.
- If $d_{ls} \geq r_m^k \geq d_{ms} \geq r_l^k$, we have $\left[\left(d_{ls} - r_m^k\right) + \left(d_{ms} - r_l^k\right)\right] - \left[\left(d_{ls} - r_l^k\right) + 0\right]$ $= d_{ms} - r_m^k \leq 0$.
- If $d_{ls} \geq r_m^k \geq r_l^k \geq d_{ms}$, we have $\left[\left(d_{ls} - r_m^k\right) + 0\right] - \left[\left(d_{ls} - r_l^k\right) + 0\right] = r_l^k - r_m^k < 0$.

- If $r_m^k \ge d_{ls} \ge r_l^k \ge d_{ms}$, we have $\left[0+0\right]-\left[\left(d_{ls}-r_l^k\right)+0\right]=r_l^k-d_{ls}\le 0$.
- If $r_m^k \ge d_{ls} \ge d_{ms} \ge r_l^k$, we have $\left[0+\left(d_{ms}-r_l^k\right)\right]-\left[\left(d_{ls}-r_l^k\right)+0\right]=d_{ms}-d_{ls}\le 0$.

We conclude that $x_{sk'}^{\text{II}}-x_{sk}^{\text{II}}\le 0$ in every case. If $\mathbf{r}^{k'}\ne\mathbf{r}^*$, set $k=k'$ and find a new permutation $\mathbf{r}^{k'}$ that satisfies conditions (5.12)-(5.14). Because $|\mathrm{K}|$ is finite and each k' indexes a different permutation of $\mathbf{r}$, the sequence of permutation indices $k,k',...,k^{(\kappa)}$ must reach $\mathbf{r}^*$ without increasing the value of x_{sk}^{II}. ∎

It follows that the permutation $\mathbf{r}^*$ minimizes the depot inventory required to satisfy any given set of scenarios D, and in particular any p_s^{II}-feasible subset. Therefore, assume that the remainder vectors $\hat{\mathbf{r}}_{sj}$ are listed in non-increasing order. We arrive at this simplified formulation **CCNIM-pIIm** ("pIIm" stands for "period II modified"):

(**CCNIM-pIIm**) $$\min_{\mathbf{w}_{sj}^{\text{II}},x_{sj}^{\text{II}}} x_{sj}^{\text{II}} \tag{5.16}$$

s. t.

$$\Pr\left\{\mathbf{w}_{sj}\ge\widetilde{\mathbf{dd}}_{sj}^{\text{II}}\right\}\ge p_s^{\text{II}} \tag{5.17}$$

$$\mathbf{1}^T\mathbf{w}_{sj}-x_{sj}^{\text{II}}=0 \tag{5.18}$$

$$\mathbf{w}_{sj}\le\overline{\mathbf{v}}-\hat{\mathbf{r}}_{sj} \tag{5.19}$$

$$\mathbf{w}_{sj}\in\mathbb{Z}_+^n \tag{5.20}$$

where $$\widetilde{\mathbf{dd}}_{sj}^{\text{II}}\equiv\left(\max\left\{\tilde{\mathbf{d}}_s^{\text{II}},\underline{\mathbf{v}}\right\}-Y_{\text{MAP}}^{\text{II}}\left(\hat{\mathbf{r}}_{sj}\right)\hat{\mathbf{r}}_{sj}\right)^+ \tag{5.21}$$

and $$Y_{\text{MAP}}^{\text{II}}\left(\mathbf{r}\right)\in\arg_{Y\in\mathrm{Y}}\left\{\left(Y\mathbf{r}\right)_i-\left(Y\mathbf{r}\right)_{i+1}\ge 0\quad\forall i\le n-1\right\} \tag{5.22}$$

CCNIM-pIIm is a simple PIP, whose solution is given by finding the PEP $\mathbf{w}_{sj}^*$, such that $x_{sj}^{\text{II}}=\mathbf{1}^T\mathbf{w}_{sj}$ is minimal. There are two simple ways to find $x_{sj}^{\text{II}*}$, using the "forward enumeration" described by Beraldi and Ruszczyński [2001] (hereafter referred to as *PEP enumeration*), and the MSP algorithm (hereafter referred to as *MSPA*), described in Kress et al. [2004].

The forward enumeration begins by obtaining a vector, which bounds the PEPs using the marginal distributions of the requirements from below. This lower bound is set as the root node in a PEP search tree. From each node in the tree, we generate the next level of nodes by generating all candidate allocation vectors that have one additional missile in one component. If a candidate allocation satisfies the demand vectors with the required probability, it is a PEP, and no more levels are generated from it.

MSPA begins by ordering all scenarios by their aggregate demand in non-decreasing order. An upper bound on the value of x_{sj}^{II} is obtained from the allocation required to satisfy the p-feasible subset defined by the first scenarios (in the aggregate ordering) whose aggregate probability of occurrence satisfies *p*. Any scenario which requires more missiles to be satisfied cannot be part of the optimal *p*-feasible subset, and is eliminated. If the aggregate probability of the remaining scenarios exceeds the threshold, MSPA generates, based on single-ship criteria, a list of scenarios that might not be included in an optimal *p*-feasible subset. MSPA then enumerates the *p*-feasible subsets resulting from elimination of appropriately weighted sets of scenarios from that list. Because MSPA typically enumerates only a small fraction of all few *p*-feasible subsets, it may be viewed as an advanced version of **CCNIM-e**.

Clearly, **CCNIM-pIIm** can be formulated as an IP and solved by an LP-based branch-and-bound solver. KPP indicate that, for a very large number of scenarios, branch-and-bound is more efficient than MSPA, but as we show later in this chapter, under those circumstances, PEP enumeration is also much more efficient, so it is unlikely that an IP would be needed. If the problem becomes very large, then implementing some of the specialized decomposition procedures described in Chapter I may be worthwhile. This dissertation focuses on the decomposition properties of **CCNIM-MAP**, which are independent of the solution method for **CCNIM-pIIm**. We believe that the simpler algorithms are more efficient for problems of practical size, and so restrict our computational analyses to those algorithms. Another advantage of using MSPA or PEP enumeration is that we can solve **CCNIM-MAP** without requiring specialized solver like CPLEX.

B. ALGORITHM

Because of the differences between **CCNIM-MAP** and **CCGIM**, and because we seek solutions for a greater range of cost ratios, a simple decomposition procedure, as used by KPP, does not guarantee we obtain an optimal solution. As shown in section A of Chapter IV, the point exposed by following their decomposition procedure, $\mathbf{x}_a$, may not be optimal for some $c_2 \le c_1$, and is certainly not optimal when $c_2 > c_1$ (except for the degenerate case where there is only one feasible solution.) We therefore need a more extensive list of period-I candidate allocations.

We refer to the algorithm that solves **CCNIM-MAP** by **CCNIM-dc** ("dc" stands for "decomposition"). **CCNIM-dc** has two distinct parts. The first part optimally solves **CCNIM** in the case that $c_1 \ge c_2$. If $c_1 < c_2$, we invoke the second part of **CCNIM-dc** to calculate $\mathbf{x}_d$, the optimal solution for cases where $c_1 \ll c_2$. This solution may not be optimal, but by setting $c_1 = c_2$ and invoking the first part of **CCNIM-dc**, we can verify its optimality or obtain a lower bound on the optimal solution value. If $c_1 \ne c_2$, we obtain every feasible period-I allocation plan that corresponds to the identified solutions.

The difference between the two parts of **CCNIM-dc** lies solely in the way in which period-I candidate allocations are generated. Theorem 2 proves that if $c_1 \ge c_2$, some period-I allocation which is a PEP of the distribution of the scenario requirements leads to an optimal solution of **CCNIM**. Furthermore, if $c_1 > c_2$, then every optimal allocation must be a PEP of that distribution. Therefore, we need only consider PEPs of the distribution of the scenario requirements as potential period-I allocations. The set of PEPs is a small subset of all potentially legal allocations, so this enumeration scheme is relatively efficient.

Theorem 2.

Let $\mathrm{P}_1 = \left\{ \mathbf{v}_1^{\mathrm{I}}, \ldots, \mathbf{v}_{|\mathrm{P}|}^{\mathrm{I}} \right\}$ be the set of all PEPs on the distribution of $\widetilde{\mathbf{dd}}^{\mathrm{I}}$, and let P be the set of period-I allocations feasible in **CCNIM-MAP**; $\mathrm{P}_1 \subseteq \mathrm{P}$ by definition. Let

$z(\mathbf{v}) \equiv c_1 \mathbf{1}^T \mathbf{v} + c_2 x^{\mathrm{II}}(\mathbf{v})$ be the total cost of a procurement plan as a function of the period-I allocation, and let $\mathbf{v}' \in \mathrm{P}$ be an arbitrary feasible allocation plan. Then,

(1) If $c_1 \geq c_2$, $\min_{\mathbf{v} \in \mathrm{P}_1} z(\mathbf{v}) = \min_{\mathbf{v}' \in \mathrm{P}} z(\mathbf{v}')$.

(2) If $c_1 > c_2$, $\min_{\mathbf{v} \in \mathrm{P}_1} z(\mathbf{v}) < \min_{\mathbf{v}' \in \mathrm{P}-\mathrm{P}_1} z(\mathbf{v}')$.

Proof:

Let $\mathbf{v}' \in \operatorname{argmin}_{\mathbf{v} \in \mathrm{P}} z(\mathbf{v})$. If $\mathbf{v}' \in \mathrm{P}_1$ then result (1) is trivial. Otherwise, there must be some $\mathbf{v}_j^{\mathrm{I}} \in \mathrm{P}_1$ such that $\mathbf{v}' \geq \mathbf{v}_j^{\mathrm{I}}$; denote $\mathrm{S}_j^{\mathrm{I}} = \{ s \in \mathrm{S}^{\mathrm{I}} \mid \mathbf{v}_j^{\mathrm{I}} \geq \mathbf{d}_s^{\mathrm{I}} \}$ as the index set of scenarios satisfied by $\mathbf{v}_j^{\mathrm{I}}$.

For every $s \in \mathrm{S}_j^{\mathrm{I}}$, the vector of remainders maintains $\mathbf{r}_s(\mathbf{v}_j^{\mathrm{I}}) \equiv (\mathbf{v}_j^{\mathrm{I}} - \mathbf{d}_s^{\mathrm{I}})^+ = \mathbf{v}_j^{\mathrm{I}} - \mathbf{d}_s^{\mathrm{I}}$. Because $\mathbf{v}' \geq \mathbf{v}_j^{\mathrm{I}}$, it is also true that $\mathbf{r}_s(\mathbf{v}') = \mathbf{v}' - \mathbf{d}_s^{\mathrm{I}}$, and $\mathbf{r}_s(\mathbf{v}') - \mathbf{r}_s(\mathbf{v}_j^{\mathrm{I}}) = \mathbf{v}' - \mathbf{v}_j^{\mathrm{I}} \geq 0$. Thus, every missile above the level $\mathbf{v}_j^{\mathrm{I}}$ is carried over into period II for every scenario $s \in \mathrm{S}_j^{\mathrm{I}}$, and could have been placed in the depot inventory, rather than on a ship. This would have reduced the total cost by $c_1 - c_2$ per missile.

Let $\overline{\mathrm{S}}_j^{\mathrm{I}}$ denote the complement of the set $\mathrm{S}_j^{\mathrm{I}}$, and suppose that for some scenario $s' \in \overline{\mathrm{S}}_j^{\mathrm{I}}$, $v_{ij}^{\mathrm{I}} - d_{is}^{\mathrm{I}} < 0$ for some ship *i*. In that case, if $v_i' > v_{ij}^{\mathrm{I}}$, then up to $d_{is}^{\mathrm{I}} - v_{ij}^{\mathrm{I}}$ more missiles would be expended in period I and not be carried over into period II. These "wasted" missiles would still have to be allocated out of the depot in period II, if scenario s' were to occur.

We conclude that $x^{\mathrm{II}}(\mathbf{v}') \geq x^{\mathrm{II}}(\mathbf{v}_j^{\mathrm{I}}) - \sum_i (v_i' - v_{ij}^{\mathrm{I}})$, which leads to:

$$z(\mathbf{v}') - z(\mathbf{v}_j^{\mathrm{I}}) = c_1 \sum_i (v_i' - v_{ij}^{\mathrm{I}}) + c_2 \left(x^{\mathrm{II}}(\mathbf{v}') - x^{\mathrm{II}}(\mathbf{v}_j^{\mathrm{I}}) \right)$$

$$\geq (c_1 - c_2) \sum_i (v_i' - v_{ij}^{\mathrm{I}}) \geq 0. \tag{5.23}$$

By assumption, $z(\mathbf{v}') \le z(\mathbf{v}_j^{\mathrm{I}})$, which, combined with (5.23), proves claim (1). Claim (2) follows from the fact that if $\mathbf{v}' \in \mathrm{P} - \mathrm{P}_1$, then $\sum_i (v_i' - v_{ij}^{\mathrm{I}}) > 0$, and so if $c_1 > c_2$, inequality (5.23) is strict. ∎

We enumerate the entire set of PEPs using the forward enumeration scheme detailed in Beraldi and Ruszczyński [2002], and calculate the objective function $z_j = c_1 \mathbf{1}^T \mathbf{v}_j^{\mathrm{I}} + c_2 x^{\mathrm{II}}(\mathbf{v}_j^{\mathrm{I}})$ associated with each one (recall that $x^{\mathrm{II}}(\mathbf{v}_j^{\mathrm{I}})$ denotes the depot inventory resulting from period-I allocation $\mathbf{v}_j^{\mathrm{I}}$). The optimal solution is given by obtaining $j_{\min} \in \operatorname{argmin}_j z_j$, and retrieving the appropriate values for the period-I allocation and depot inventory.

Note that an alternative approach would enumerate all p^{I}-feasible subsets of the period-I scenarios and calculate the minimal allocations to satisfy those subsets. Perhaps some candidate subsets could be eliminated, but we have not explored this possibility. This scheme may be more efficient if the demand values are themselves large, if the number of p-feasible subsets is small, or if the probability threshold is high. We do not expect to encounter these conditions in most practical applications of **CCNIM-MAP**, because the demands are usually bounded, and there are more than a handful of scenarios in each period.

If $c_1 < c_2$, then, as discussed in Chapter IV, we seek the point $\mathbf{x}_d$, which minimizes the depot inventory. To find $\mathbf{x}_d$, we initialize the set of potentially optimal allocations $\mathrm{A} = \{\mathbf{v}_1^{\mathrm{I}}\}$, where $\mathbf{v}_1^{\mathrm{I}} = \overline{\mathbf{v}}$, and calculate $x_d^{\mathrm{II}} \equiv x^{\mathrm{II}}(\mathbf{v}_1^{\mathrm{I}})$ by solving the appropriate instance of **CCNIM-pIIm**. We then begin an iterative trial-and-error method to find the minimum period-I allocation that yields x_d^{II}, thus obtaining $\mathbf{x}_d$. From the allocation in A, we create a list of predecessors, which are the allocations that have one missile less in exactly one ship's inventory. Because the demands in every scenario are ordered, we only need to generate predecessors that maintain the ordering $u_i \ge u_{i'}$ for

$i < i'$. Therefore, from $\mathbf{v}_1^{\mathrm{I}}$ we generate only one predecessor, in which $v_{1i}^{\mathrm{I}} = \bar{v}$ for $i < n$ and $v_{1n}^{\mathrm{I}} = \bar{v} - 1$. We refer to predecessors that are p^{I}-feasible as *candidates*, and eliminate those that are not.

We use the following enumeration scheme, adapted from the backwards enumeration scheme proposed by Beraldi and Ruszczyński [2001]. In order to avoid creating duplicate predecessors, this scheme relies on the fact that $u_i \geq u_{i'}$ $i < i'$ in any candidate allocation. With each potential allocation in $\mathtt{A}$, we maintain δ_j, the index of the ship where allocation *j* differs from its successor. For each $\mathbf{v}_j^{\mathrm{I}} \in \mathtt{A}$, we generate only predecessors that reduce the inventory to either the ship indexed by $\delta_j - 1$ or by δ_j.

To check p^{I}-feasibility efficiently, we first calculate $\boldsymbol{\beta}^{\mathrm{I}}$, the minimum allocation plan that satisfies every period-I scenario. Clearly, if $\mathbf{u}_j \geq \boldsymbol{\beta}^{\mathrm{I}}$, then it must be p^{I}-feasible. If this test is not passed, a rigorous check for feasibility is performed. Note that it is possible that $x_d^{\mathrm{I}} \ngeq \boldsymbol{\beta}^{\mathrm{I}}$, because some scenario may have, for example, $d_{is}^{\mathrm{I}} = \bar{v}_i \ \forall i$, and never be satisfied by an optimal plan.

We calculate the induced depot inventory level $x^{\mathrm{II}}\left(\mathbf{u}_j\right)$ for all of the candidate allocations. Any candidate inducing a depot level of x_d^{II} missiles is an improvement over the allocations of the previous iteration, and is potentially optimal. These potentially optimal allocations are used to create the next generation of candidate allocations. The algorithm repeats until no more legal allocations can be found, or none of the tested allocations yields a two-period solution requiring only x_d^{II} missiles in the depot. The incumbent period-I allocations all require x_d^{I} missiles, and are equally favorable.

The solution $\mathbf{x}_d$ is optimal if c_2 is sufficiently large, but calculating the minimum value of c_2 at which $\mathbf{x}_d$ is optimal is difficult. We can, however, obtain an upper bound on c_2, beyond which $\mathbf{x}_d$ is guaranteed optimal, by solving **CCNIM-dc** again, this time setting $c_1 = c_2$, and obtaining $\mathbf{x}_{\min}$. We then calculate $\Delta = \left(x_d^{\mathrm{I}} + x_d^{\mathrm{II}}\right) - \left(x_{\min}^{\mathrm{I}} + x_{\min}^{\mathrm{II}}\right)$. If

$\Delta = 0$, then $\mathbf{x}_d$ is optimal for any $c_1 < c_2$. Otherwise, a lower bound on the optimal solution is given by

$$\underline{z} = c_1\left(x_d^{\mathrm{I}} - 1 - \Delta\right) + c_2\left(x_d^{\mathrm{II}} + 1\right). \tag{5.24}$$

The threshold cost ratio for which $\mathbf{x}_d$ is guaranteed optimal is calculated such that $\underline{z} = z_d$, and is given by $c_2/c_1 = 1 + \Delta$. We define the relative gap in **CCNIM-MAP** by

$$gap = \left(z_d - \underline{z}\right)/\underline{z}. \tag{5.25}$$

A skeletal, pseudo-code description of **CCNIM-dc** is given here using vector notation, with vectors always being *n*-dimensional columns. We assume that the ships are listed in non-increasing order of their inventory lower bounds, and that their capacities (upper bounds) are equal. If the capacities are not equal, a small modification is required to ensure a feasible solution, and that solution is not guaranteed optimal for **CCNIM-MAP**. Appendix E provides a more detailed description of **CCNIM-dc**.

Data

$i \in \{1,...,n\}$	ships
$\mathrm{S}^{\mathrm{I}} = \left\{s_1,...,s_{\lvert \mathrm{S}^{\mathrm{I}} \rvert}\right\}$	period-I scenarios.
$\mathrm{S}^{\mathrm{II}} = \left\{s'_1,...,s'_{\lvert \mathrm{S}^{\mathrm{II}} \rvert}\right\}$	period-II scenarios.
$\mathbf{d}_s^{\mathrm{I}}$	vector of demands for period I scenario *s*
$\mathbf{d}_{s'}^{\mathrm{II}}$	vector of demands for period II scenario s' (Each demand vector is preordered in non-increasing order.)
p^{I}	probability threshold for period I
p_s^{II}	probability threshold for period II, following period-I scenario *s*
φ_s	probability of period-I scenario s
$\varphi_{s'\mid s}$	conditional probability of period-II scenario s' on period-I scenario *s*
$\overline{\mathbf{v}}$	missiles capacity (upper bound, for each ship, on the number of missiles it can carry.) Assume $\overline{v}_i = \overline{v}$ for all *i*.
$\underline{\mathbf{v}}$	discretionary lower bounds on missile load-outs. Assume bound vectors are non-increasing.
c_1	cost of allocating a missile to a ship
c_2	cost of allocating a missile to the depot

```
(1)  Algorithm CCNIM-dc
(2)  begin
(3)    if $c_1 \ge c_2$ then                    // Part 1
(4)      for $s := 1$ to $|\mathtt{S}^{\mathrm{I}}|$ do
(5)        $\mathbf{dd}_s^{\mathrm{I}} := \max\{\mathbf{d}_s^{\mathrm{I}}, \underline{\mathbf{v}}\}$;   // number of missiles required in successful scenarios
(6)      end;
(7)      GENERATE $\mathtt{A} := \{\mathbf{v}_1^{\mathrm{I}}, ..., \mathbf{v}_{|\mathtt{A}|}^{\mathrm{I}}\}$, all PEPs of $\mathbf{dd}_s^{\mathrm{I}}$;   // see theorem 2
(8)      for $j := 1$ to $|\mathtt{A}|$ do                  // allocation index
(9)        for $s := 1$ to $|\mathtt{S}^{\mathrm{I}}|$ do
(10)         $\mathbf{r}_{sj} := \left(\mathbf{v}_j^{\mathrm{I}} - \mathbf{d}_s^{\mathrm{I}}\right)^+$;          // remaining inventories following scenario s
(11)         $\mathbf{r}_{sj} := \mathrm{SORT}(\mathbf{r}_{sj}, \text{'descend'})$;        // by inventory size
(12)         $\mathbf{dd}_{s'|s,j}^{\mathrm{II}} := \left(\max\{\mathbf{d}_{s'}^{\mathrm{II}}, \underline{\mathbf{v}}\} - \mathbf{r}_{sj}\right)^+$;     // period-II requirement vector
(13)         $w_{sj}^{\mathrm{II}} := \min_{\mathbf{w}}\{\mathbf{1}^T \mathbf{w}\}$
(14)                 s.t.   $\mathbf{w}$ is a $p_s^{\mathrm{II}}$-efficient point of $\mathbf{dd}_{s'|s,j}^{\mathrm{II}}$;
(15)       end;
(16)       $x_j^{\mathrm{II}}\left(\mathbf{v}_j^{\mathrm{I}}\right) := \max_s w_{sj}^{\mathrm{II}}$;      // we refer to lines (9)-(16) as calculating $x_j^{\mathrm{II}}\left(\mathbf{v}_j^{\mathrm{I}}\right)$
(17)     end;
(18)     $\hat{j} := \operatorname{argmin}_j\{c_1 \mathbf{1}^T \mathbf{v}_j^{\mathrm{I}} + c_2 x_j^{\mathrm{II}}\}$;   $\hat{\mathbf{v}}^{\mathrm{I}} := \mathbf{v}_{\hat{j}}^{\mathrm{I}}$;   $\hat{x}^{\mathrm{II}} := x_{\hat{j}}^{\mathrm{II}}$;
(19)   else                                   // Part 2
(20)     $\mathtt{C} := \{\overline{\mathbf{v}}\}$;                          // initialize current set of candidates
(21)     $\delta = \{n\}$;                          // initialize predecessor index
(22)     $\mathtt{CN} := \varnothing$;                        // initialize set of candidates for next generation
(23)     $x_d^{\mathrm{II}} := x^{\mathrm{II}}(\overline{\mathbf{v}})$;                 // see Part 1
(24)     while $\mathtt{C} \ne \varnothing$
(25)       for $j := 1$ to $|\mathtt{C}|$ do
(26)         for $j' := \max\{\delta_j - 1, 1\}$ to $\delta_j$ do
(27)           $\mathbf{u} := \mathtt{C}_j$;
(28)           $u_{j'} := u_{j'} - 1$;                 // reduce one component only
(29)           if ($\mathbf{u}$ is $p^{\mathrm{I}}$-feasible) and $\left(u_{j'-1} \ge u_{j'} \ge u_{j'+1}\right)$ then
(30)             $\mathtt{CN} := \mathtt{CN} \cup \{\mathbf{u}\}$;  $\delta\mathtt{N} := \delta\mathtt{N} \cup \{j'\}$;
(31)           end;
(32)         end;
(33)       end;
```

```
(34)            for $j := |\mathtt{CN}|$ to 1 by $-1$ do
(35)                if $x_j^{\mathrm{II}}(\mathbf{u}_j^{\mathrm{I}}) > x_d^{\mathrm{II}}$ then
(36)                    $\mathtt{CN} := \mathtt{CN} \backslash \{\mathbf{u}_j^{\mathrm{I}}\}$; $\delta\mathtt{N} := \delta\mathtt{N} \backslash \{\delta\mathtt{N}_j\}$;
(37)                end;
(38)            end;
(39)            if $\mathtt{CN} == \varnothing$ then
(40)                $\hat{\mathtt{A}} := \mathtt{C}$;
(41)            end;
(42)            $\mathtt{C} := \mathtt{CN}$; $\delta := \delta\mathtt{N}$;
(43)        end;
(44)        $\hat{\mathbf{v}}^{\mathrm{I}} := \mathbf{v}_j^{\mathrm{I}} \in \mathtt{A}$; $\hat{x}^{\mathrm{II}} := x_d^{\mathrm{II}}$;
(45)    end;
(46) end;
```

C. COMPUTATIONAL RESULTS

1. Comparison with RFFAM

To test the performance of **CCNIM-dc**, we repeat each case reported in Chapter II. The algorithm is implemented in Matlab 7.1, and run on a 2.8 GHz personal computer under the Microsoft XP operating system. Because the run times are very short compared with the timekeeping resolution ($1/64$ seconds), we measure the time required to solve 1000 replications of each instance, and divide the result by 1000. The algorithm expends the same amount of effort for any ratio $c_2/c_1 \leq 1$, and the same amount of effort when $c_2/c_1 > 1$. The results are summarized in Table 5. For cases with $c_2/c_1 \leq 1$, we also report the number of dominating period-I allocations that are actually examined by the algorithm. For cases with $c_2/c_1 > 1$, we also report the number of times the algorithm solved a period-II problem.

The ability of the Matlab code to solve all of the specified cases in a few milliseconds is very encouraging. We perform more extensive testing of the computational behavior by randomly generating instances for cases of various sizes. Solving **CCNIM-dc** when $c_2/c_1 \leq 1$ is a two-step procedure, and we investigate the behavior of each step separately. In particular, we compare the performance of the two methods available to solve **CCNIM-pIIm**, namely PEP enumeration (until the first PEP is found) and MSPA. This analysis will enable us to choose the appropriate algorithm based on a problem's characteristics. When $c_2/c_1 > 1$, we are only solving instances of the period-II problem while enumerating period-I allocations of interest. In this case, the algorithm's computation time is, therefore, directly proportional to the time required to solve a single period-II instance of that size.

	$c_2/c_1 \le 1$		$c_2/c_1 > 1$	
Case Name	Number of Period-I PEPs	Average CPU-Time (msec.)	Number of Period-II Problems	Average CPU-Time (msec.)
case 2a	2	3.9	2	2.2
case 2b	2	3.8	2	2.4
case 2c	2	5.8	2	4.6
case 2d	1	2.3	2	4.1
case 2f	2	6.7	2	6.4

Table 5. **CCNIM-dc** Solution Results for Specified Cases.

"Case Name" references the cases defined for RFFAM; see Appendix C for details. "Average CPU-time" is the average time over 1000 runs required to optimally solve the instance, in milliseconds. "Number of Period-I PEPs" is the number of period-I allocations for which the two-period solution must be calculated to determine the optimal solution when $c_2/c_1 = 1$. "Number of period-II problems" specifies the number of times a period-II problem was solved. This includes the initial solution of x_d^{II}. Solution times are extremely short, even for case 2f, for which **RFFAM** cannot obtain an integer solution in one hour of computation.

2. Period-I Solution Times

Here, we randomly generate 100 instances for each case size; all demands for missiles are uniformly distributed integers on [0,8]. Allocation lower and upper bounds are set at 2 and 8 missiles, respectively, for all ships. We assume every scenario is equi-probable. We observe the performance of the PEP enumeration as each of the three parameters, n, $\left|S^{\mathrm{I}}\right|$, and p^{I}, varies separately. Because solution times may be very short compared with the resolution of the timekeeping mechanism, we repeatedly solve each instance until a discretization error of less than 1% is achieved (a minimum runtime of 1.56 seconds), and report the observed time divided by the number of times the enumeration algorithm solved.

It is important to stress that the specific results are strongly affected by the distribution of the generated scenarios. The purpose of these trials is only to observe general trends, and verify that problems of useful size can be solved in a reasonable length of time.

We first observe the effect of the number of ships on the enumeration time of the entire set of PEPs of the period-I requirements vectors. The number of period-I scenarios is set at $\left|S^{\mathrm{I}}\right| = 12$, and the probability threshold is set at $p^{\mathrm{I}} = 0.75$. The distribution of the computation times is summarized in the left graph of Figure 3, which depicts the median, the quartiles and the minimum and maximum observed times. The graph on the right of Figure 3 displays the same statistics for the *enumeration gap*, i.e., the difference between the maximum level of any PEP of the requirements vector and the level of the PEP lower bounding vector. (Recall that the "level" of an integer vector is defined as the sum of its components.)

There are two of observations we wish to emphasize. First, note that every statistic except for the minimum time increases exponentially as the number of ships increases. The minimum value appears to increase only linearly, if we ignore the sample for $n = 24$. Second, it is clear that there is great variability in the computation times for cases of equal size, and that this variability increases as n increases. For cases of 20 ships or more, the computation time of the entire set of PEPs ranges over three orders of magnitude. This variability makes it difficult to predict, based on the case size alone, how difficult solving a specific case will be. Clearly, the easiest cases involving 28 ships require less effort than the hardest cases involving as few as 8 ships.

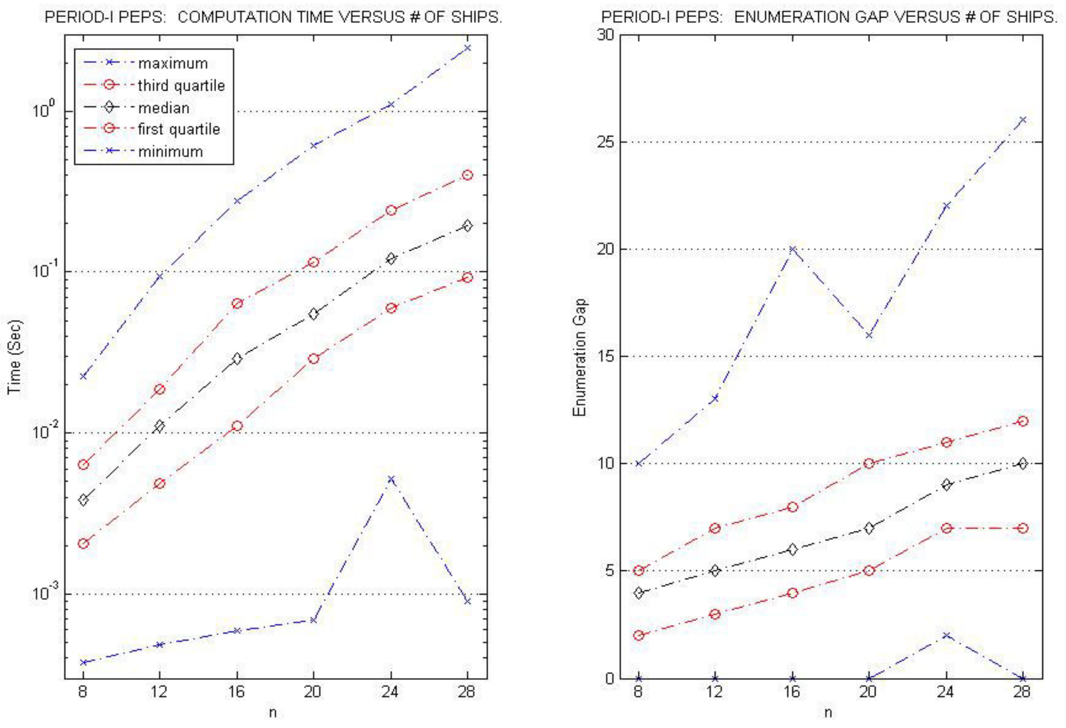

Figure 3. Period-I PEP enumeration Versus Number of Ships. The graph on the left presents the median (diamonds), first and third quartiles (circles), and minimum and maximum values (crosses) of the computation times of the PEPs. The computation times are drawn on a logarithmic scale, as *n*, the number of ships, varies linearly. We set $\left|S^{\mathrm{I}}\right| = 12$ and $p^{\mathrm{I}} = 0.75$. The graph on the right presents the same statistics for the enumeration gap, but on a linear scale. The PEP enumeration time is strongly correlated with the enumeration gap.

The Spearman's rank correlation [e.g., Conover 1999, pg. 314] between the computation time and the enumeration gap, taken over 600 observations, is 0.8127 with a p-value of practically zero. It is no surprise that the PEP enumeration time is strongly correlated with the enumeration gap because the enumeration procedure generates a search tree, whose depth equals that gap. Any node in the tree, which does not represent a PEP, contributes up to *n* successors, depending on the number of monotonic vectors, which conform to the capacity bounds, that can be generated by adding 1 to any single component of the vector represented by that node. Therefore, if we denote the enumeration gap by *g*, the number of nodes in the tree is $\mathrm{O}\left(n^{g}\right)$.

The Spearman's correlation is fundamental to understanding the behavior of the computation time as a function of different case parameters, as we show in the following figures. To understand how changes in some case-size parameters affect the enumeration time, it suffices to understand how they affect the enumeration gap. For some parameters this may be easier than for others, but in any case, this behavior is, to some extent, an artifact of the scenario-generation scheme.

From Figure 3, it seems that the majority of enumeration gaps increase linearly with *n*. Intuitively, this can be explained as follows. The PEP lower bound is constructed from the marginal distribution of each element in the requirements vector, based on all of the scenarios. There is a positive probability that the value of the PEP exceeds the lower bound for each element (except those equaling the capacity bound). As the number of ships increases, and the requirement vectors contain more elements, the expected number of PEP elements that exceed their lower bound increases, contributing to the observed increase in the enumeration gap.

We next observe the behavior of the PEP enumeration algorithm as the number of scenarios varies, as shown in Figure 4. There are 12 ships in each scenario, and the probability threshold set at $p^{\mathrm{I}} = 0.75$. We observe a slow increase in the enumeration gap, which is responsible for the increase in computation time. Again, the reason for this increase is an artifact of the scenario-generation mechanism, and does not necessarily reflect on problem size. Intuitively, the PEP lower-bounding vector has some asymptotic value, and, as the number of scenarios increases, the computed lower bound approaches this asymptotic vector. However, as the number of scenarios increases, the number of p-feasible subsets increases exponentially, and the likelihood of there being more than a few PEPs increases. Then, the probability that some PEP would be of a high level increases, increasing the gap. These effects can be seen in Figure 5.

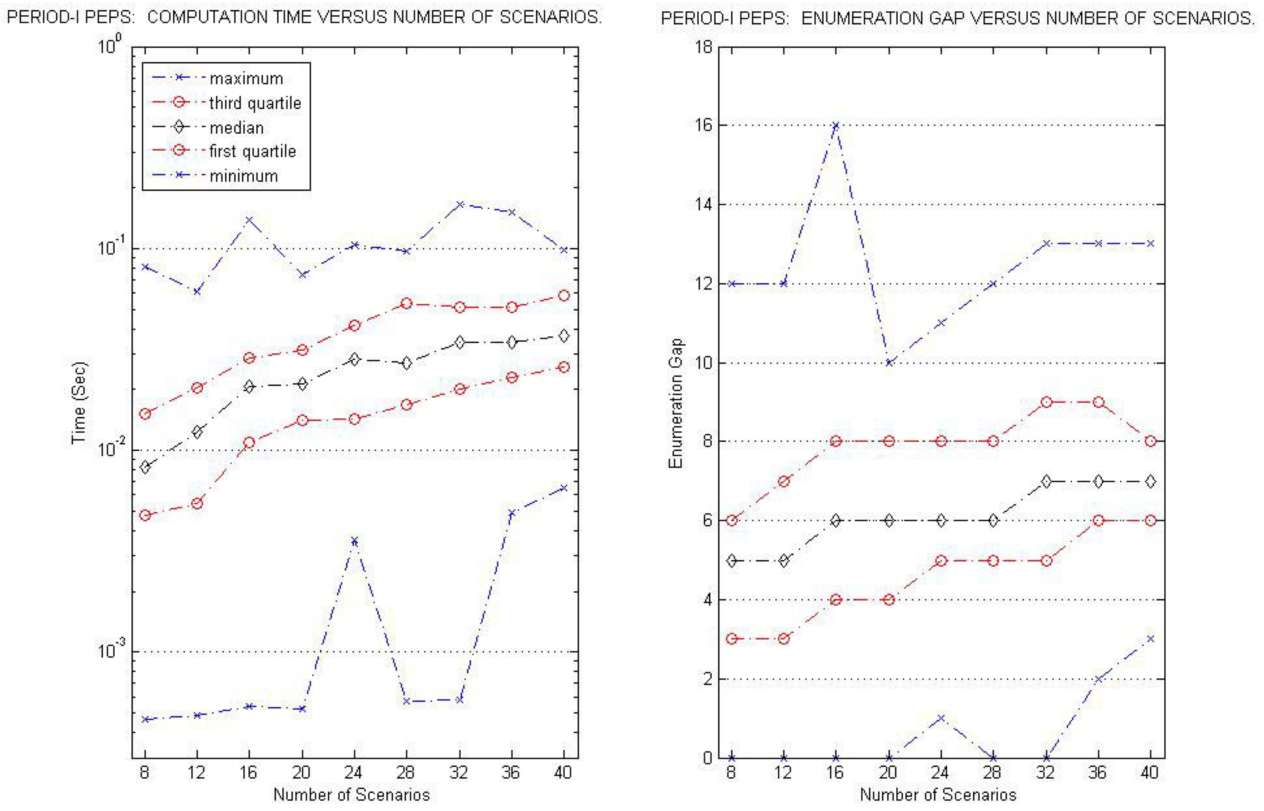

Figure 4. Period-I PEP enumeration Versus Number of Scenarios.
The statistics presented are the same as in Figure 3. We vary the number of scenarios, and set $n = 12$ and $p^{\mathrm{I}} = 0.75$.

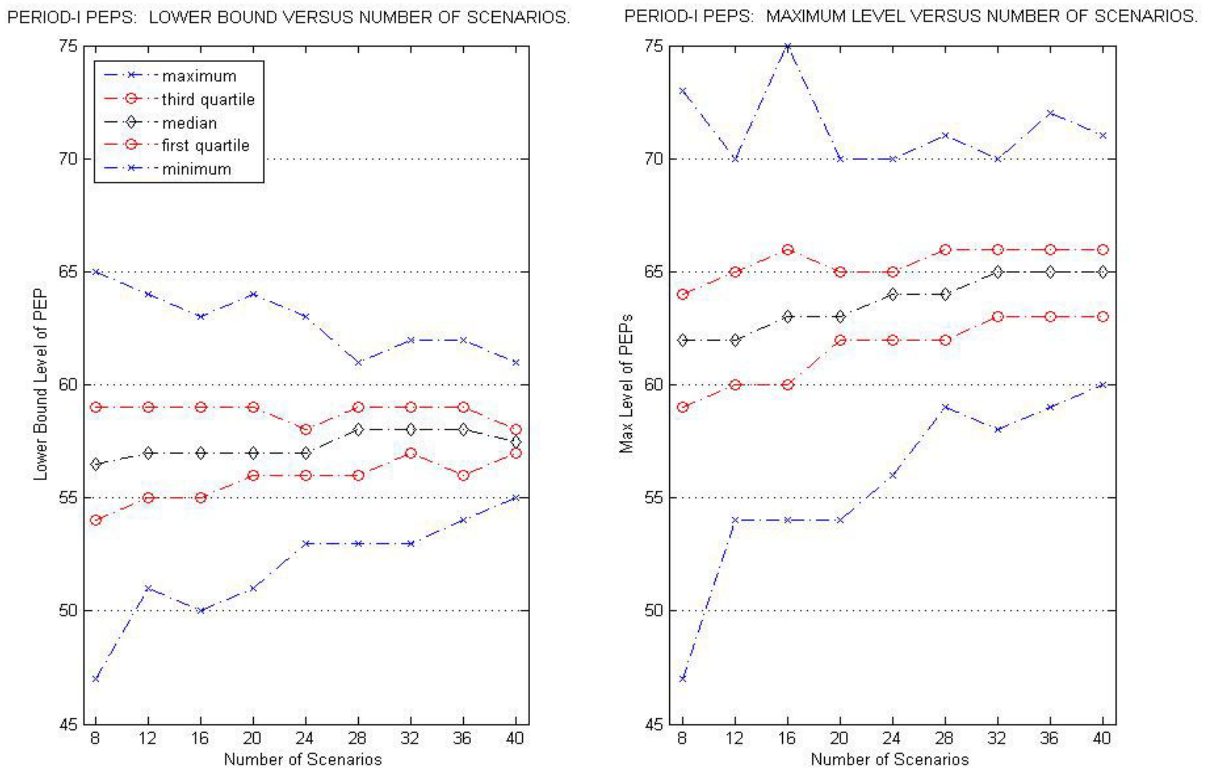

Figure 5. PEP Lower Bound Level Compared with Maximum Level. As the number of scenarios increases, the PEP lower bound level converges to some asymptotic value, but the maximum level appears to increase slowly. This explains the increase in the enumeration gap, observable in Figure 4.

Finally, we observe the performance of the enumeration procedure as the probability threshold varies. For 12 ships and 10 scenarios, we vary the probability threshold in steps of 0.1, so that at each observed probability, the size of a p-feasible subset changes by one. As shown in Figure 6, decreasing the probability threshold tends to increase the enumeration gap, and as a consequence, the computation time as well. Lower probability thresholds tend to have a larger gap because the PEP lower-bound level tends to decrease faster than the PEP maximum level. At the low extreme, the PEP lower bounding vector is the 10% sample quantile marginal distributions, and the maximum PEP is the largest level of the 10 vectors. At the other extreme, the lower bounding vector is the 100% quantile, there is only one PEP, comprising the maximum over all 10 vectors, and the two values coincide.

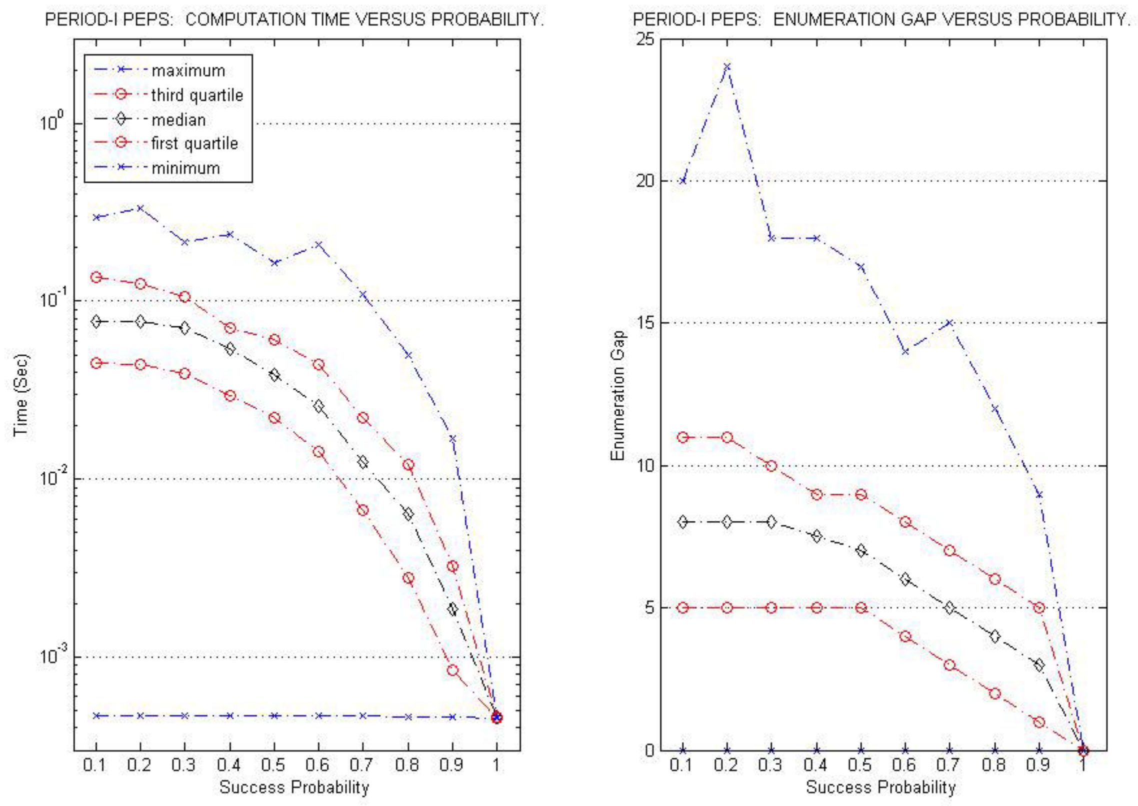

Figure 6. Period-I PEP Enumeration Versus Success Probability. The statistics presented are the same as in Figure 3. In each case, we set $n = 12$ and $|S^{\mathrm{I}}| = 12$. The probability threshold varies between 0.1 and 1. At lower success probabilities, the enumeration gap tends to be larger, resulting in longer computation times.

In summary, the enumeration of Period-I PEPs does not require much computation effort, and can be completed in about one second of computation for cases of practical size. The Period-I enumeration need not be, therefore, an issue of concern in practical applications.

3. Period-II Solution Times

We now consider the computation time for the depot inventories, required for period II. In order to solve **CCNIM-MAP, CCNIM-dc** must compute $x^{\mathrm{II}}\left(\mathbf{v}^{\mathrm{I}}\right)$, the required depot inventory for a single period-I allocation times several times. In this section we report the times required to compute $x_j^{\mathrm{II}}\left(\mathbf{v}_j^{\mathrm{I}}\right)$, for a randomly selected period-I PEP from those obtained when $\left|\mathbb{S}^{\mathrm{I}}\right| = 12$ and $p^{\mathrm{I}} = 0.75$. Note that computing $x_j^{\mathrm{II}}\left(\mathbf{v}_j^{\mathrm{I}}\right)$ itself requires $\mathrm{O}\left(\left|\mathbb{S}^{\mathrm{I}}\right|\right)$ computation time, because the depot inventory is calculated separately for the remainders following each period-I scenario *s*. We compare the performance of PEP enumeration with that of MSPA, as a function of n, $\left|\mathbb{S}^{\mathrm{II}}\right|$, and p_s^{II}. This may help us choose the appropriate algorithm based on the size of the case we wish to solve.

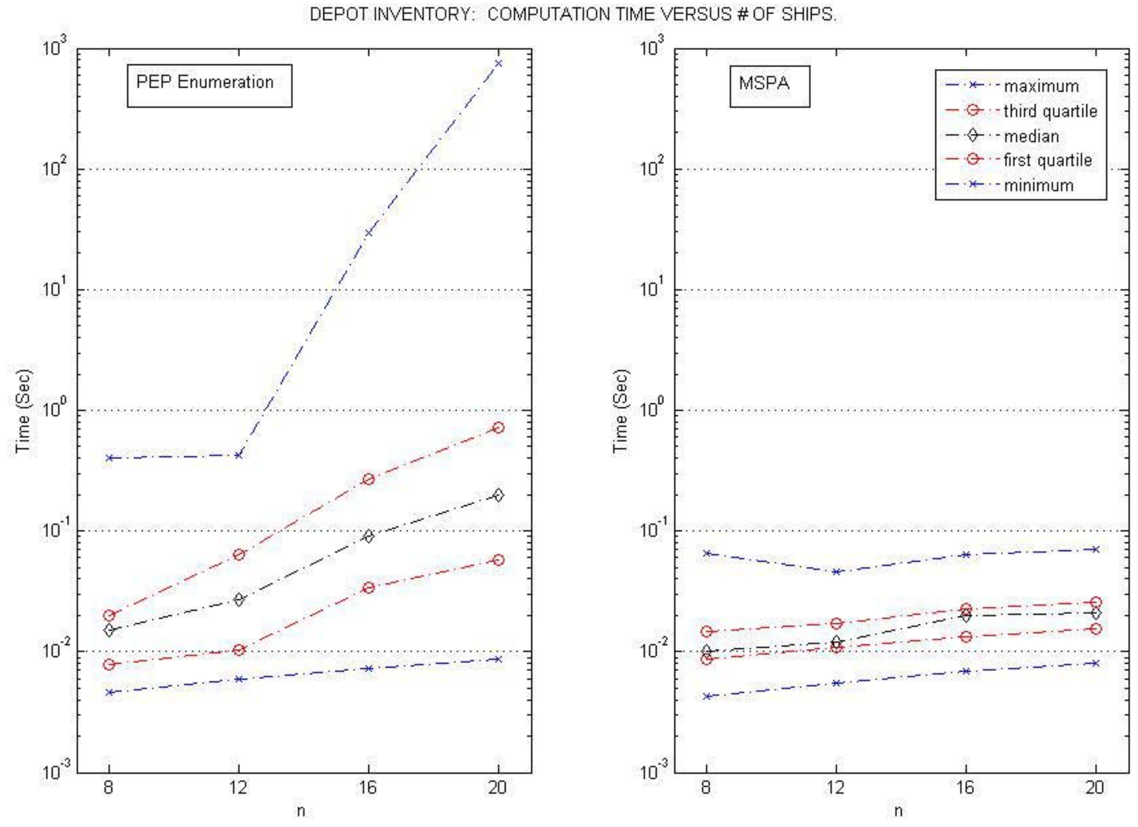

Figure 7. Depot Inventory Calculation: Time Versus Number of Ships. The statistics presented are the same as in Figure 3. In each case, $\left|\mathbb{S}^{\mathrm{II}}\right| = 12$ and $p^{\mathrm{II}} = 0.75$. The graph on the left presents the computation times achieved by PEP enumeration and the graph on the right presents the same data for computation by the MSPA. As expected, the MSPA, whose computation time is theoretically linear in *n*, is much more efficient for long demand vectors than is PEP enumeration.

We set $\left|\mathrm{S}^{\mathrm{II}}\right| = 12$ and $p^{\mathrm{II}} = 0.75$, and vary the number of ships. The graphs in Figure 7 summarize the distribution of the computation time of the optimal depot inventory for a single allocation, $x_j^{\mathrm{II}}\left(\mathbf{v}_j^{\mathrm{I}}\right)$, when it is calculated by PEP enumeration (on the left) or by MSPA (on the right). As expected, the computation time of the PEP enumeration algorithm is exponential in *n*. In contrast, the MSPA algorithm's behavior is theoretically linear in *n*, and this is corroborated by the results of Figure 7.

It is interesting to compare the PEP enumeration to the performance achieved in period I. In period II, the PEP enumeration algorithm halts once the first PEP, of minimal level, is found. However, the median computation time is 3.3 times longer than in period I, averaged over *n*, because the requirement vectors in period II are not monotonic. The gap between the PEP lower bound level and the level of the minimum PEP tends to be actually larger than the gap between the PEP lower bound level and that of the maximum PEP in period I. Furthermore, each node generates more successors, on average, because the monotonicity restriction is relaxed. A worst-case time of 750 seconds was observed, about 1000 times longer than the worst case Period-I PEP enumeration time for 20 ships.

We next compare the behavior of the two algorithms as the number of scenarios varies. We assume a combat force of 12 ships, and set the probability threshold at $p^{\mathrm{I}} = 0.75$. The computation times for both algorithms increase exponentially as the number of scenarios increases. However, the rate of increase for MSPA is greater than that for PEP enumeration. When the number of scenarios is small MSPA, is more efficient; for $\left|\mathrm{S}^{\mathrm{II}}\right| \geq 24$ it appears that PEP enumeration is faster.

We end this section by comparing the behavior of the two algorithms as the probability threshold varies. For cases with 12 ships and $\left|\mathrm{S}^{\mathrm{II}}\right| = 20$, we vary the probability threshold over the range 0.5-0.95 (we already observed in period-I that the result is trivial when $p = 1$). The results are summarized in Figure 9 in similar fashion to

Figure 7. As can be seen, there is little difference in the distribution of the computation time between the two algorithms, but MSPA seems to hold a slight advantage for all thresholds $p^{\mathrm{II}} \le 0.9$.

We believe that in most applications to naval forces, the number of ships will not exceed 20. The number of hand-generated scenarios is probably less than 10, but may be much greater if automatic scenario generation is applied. From the above analysis, it seems that the MSPA is preferable for solving the period-II depot inventory problem, with the caveat that the number of scenarios should not be too large.

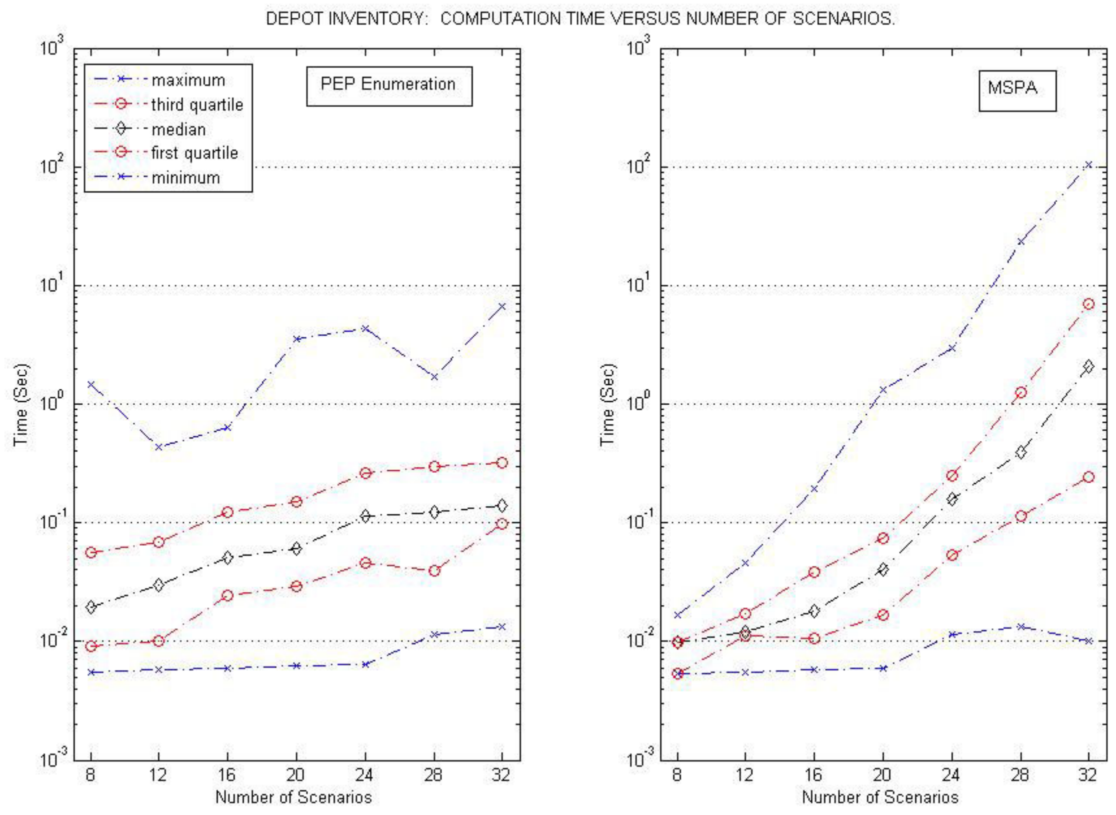

Figure 8. Depot Inventory Calculation Time Versus Number of Scenarios. The format of theses graphs is similar to that of Figure 7. We vary the number of period-II scenarios and set $n = 12$ and $p^{\mathrm{II}} = 0.75$. Both algorithms' computation time increases exponentially, but the rate of increase for MSPA is greater than for PEP enumeration. Although MSPA is more efficient for a small number of scenarios, PEP enumeration becomes more efficient if the number of scenarios exceeds 24.

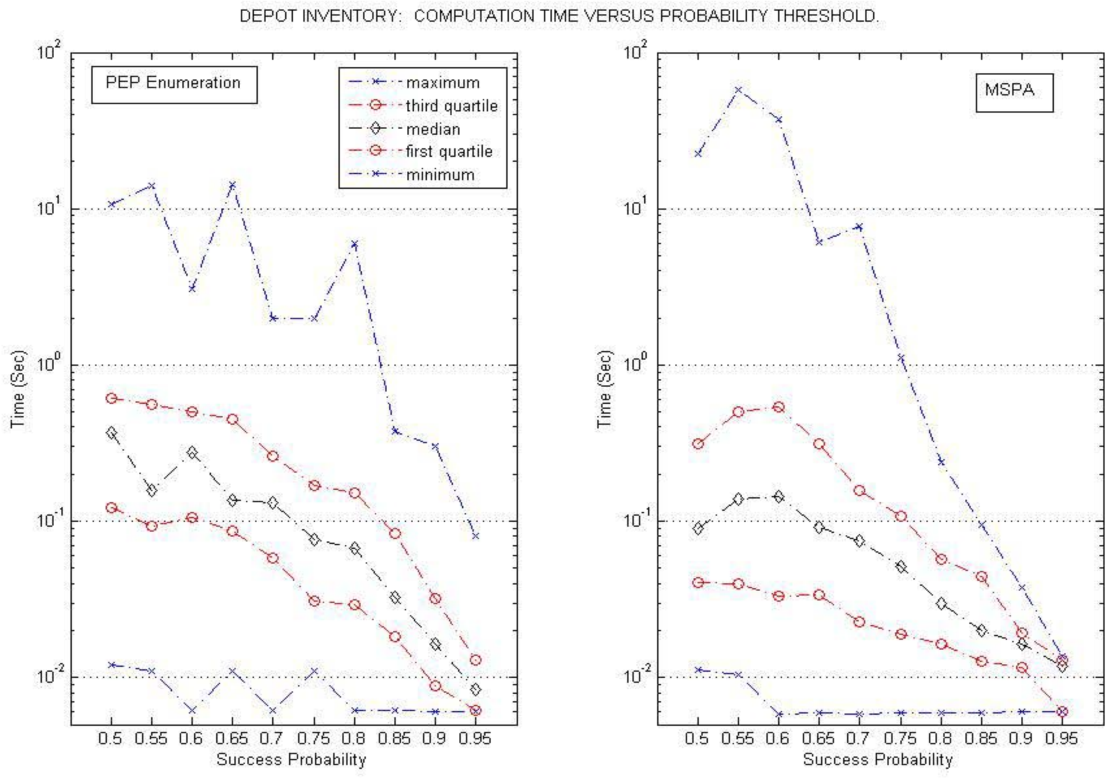

Figure 9. Depot Inventory Calculation Time Versus Probability Threshold. The format of theses graphs is similar to that of Figure 7. We vary the probability threshold for period II. and set $n = 12$ and $|S^{II}| = 20$. There is not much difference between distribution of the computation times of the depot inventory by PEP enumeration and by MSPA.

4. CCNIM-dc Solution Times

We next solve multiple instances of **CCNIM-dc**, for both cost ratios, to get a feel for solution times as case size increases. We use the MSPA to obtain the depot inventory in each case. We generate 100 instances of each case size, and maintain $n = |S^{I}| = |S^{II}|$ and $p^{I} = p_s^{II} = 0.75$. Figure 10 summarizes the distribution of computation times for $c_2/c_1 \le 1$ (left graph) and $c_2/c_1 > 1$ (right graph). The solution times increase exponentially as case size increases, which is to be expected from previous results. However, even for cases of size 24, most instances solve in less than ten seconds, and the worst-case observed is only 270 seconds. A case of that size is probably as large as we would need to solve, and the probability thresholds are not likely to be set lower.

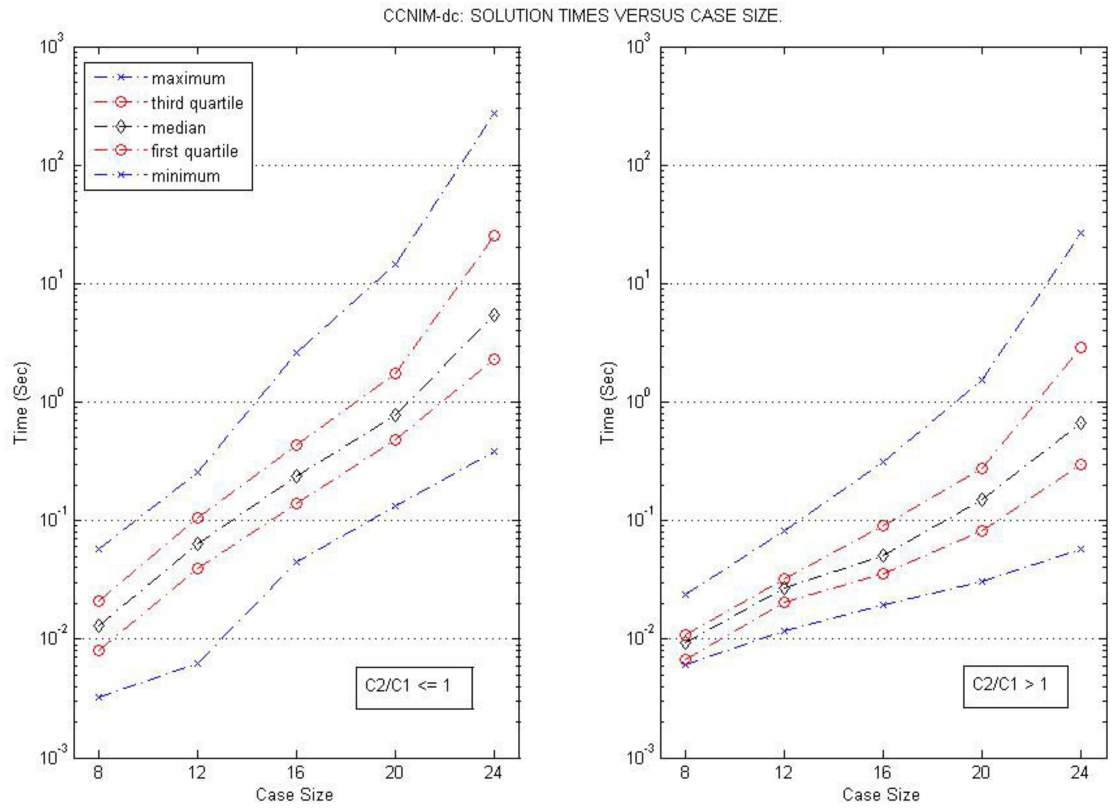

Figure 10. Distribution of **CCNIM-dc** Solution Times.
The graphs summarize the distribution of **CCNIM-dc** solution times as the case size varies, for any cost ratio; $p^{\mathrm{I}} = p_s^{\mathrm{II}} = 0.75$. As case size increases, solution times increase exponentially, and the relative dispersion of the solution times increases as well.

5. CCNIM-dc Optimality Gaps

CCNIM-dc solves optimally for most cost ratios, but if $c_2 > c_1$, an optimality gap (5.25) usually remains. The size of the optimality gap is data dependent, and instances where it is large can be created. For any given case parameters, the optimality gap is maximized when $c_2 = c_1 + \varepsilon$ and $\varepsilon \to 0$. For the random scenario generation scheme we adopt for these tests, however, the gap tends to be small. The mean relative optimality gap is roughly 5%, and the maximum, over all cases, is 18%. The equilibrium cost ratio, however, tends to increase with case size, and in some instances is as high as 21. It appears, then, that in large cases, there is a much wider range of cost ratios for which some optimality gap persists, but the gap is not necessarily larger than with smaller cases. Table 6 provides more detail.

Case Size	Mean Maximum Gap	Max Maximum Gap	Mean Threshold Cost- Ratio	Max Threshold Cost- Ratio
8	0.0583	0.1795	4.48	15
12	0.0528	0.1610	6.13	20
16	0.0449	0.1234	6.85	20
20	0.0445	0.0963	8.5	20
24	0.0409	0.0868	9.32	21

Table 6. **CCNIM-dc** Optimality Gaps for Specified Cases.

"Case Size" references the number of ships and scenarios in each period, which are all equal. The maximum optimality gap is given by $\left(z_d - \underline{z}\right)/\underline{z}$, and the threshold cost-ratio, above which $\mathbf{x}^d$ is guaranteed optimal, is given by $c_2/c_1 = 1+\Delta$. Notice that although the threshold increases significantly with case size, the optimality gap does not. In fact, both the mean and the maximum observed gaps tends to decrease with case size.

D. CONCLUSIONS

This chapter has developed **CCNIM-dc,** an algorithm to directly solve **CCNIM** without the use of an LP-based solver. **CCNIM-dc** requires that all ships have equal capacity, and yields an optimal solution for a wide range of cost ratios. Although the computation time increases exponentially with case size, these times are still quite reasonable even for large, practical problems. All cases are solved optimally except when $1 < c_2/c_1 < 1+\Delta$. But, when c_2/c_1 is within this range, the average maximum optimality gap is only about 5%.

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

We seek to determine minimum-cost initial ship load-outs plus depot-level inventory, while ensuring, with sufficiently high probability, that all ships can satisfy their assigned missions for each combat period. Each mission represents a target and its associated "demand," which is the number of missiles required to successfully engage that target. A "cost ratio," the cost of a missile stored in the depot for potential use in period II divided by those initially allocated to the ships, reflects operational preferences rather than actual monetary costs.

We initially formulate the problem as a two-period, which we denote by **CCNIM** (Chance-Constrained Naval Inventory Model). We assume that all targets will be in range of each combat ship in the fleet, and the force commander can assign any target to any ship; we denote these assignment options by the term "flexible assignments." The force commander assigns at most one target to each ship, and does so based on the available missile inventories and the target demands. We further assume that ships have bounded capacities, carry no safety stocks, are not recalled to port to offload excess inventories prior to period II, and there is no direct transshipment of missiles between ships. Due to the flexibility in target assignments inherent in naval combat, the natural model, **CCNIM**, involves three stages, and is computationally difficult to solve.

We show that by following a monotonic assignment policy (MAP), under which targets with larger demands are assigned to ships with larger inventories, the commander will satisfy every scenario possible with his current supply of missiles. And, because this policy is reasonable in the heat of battle, we assume the commander will follow it. Given the MAP, we can reduce **CCNIM** to a two-stage stochastic program, **CCNIM-MAP**, which is easier to solve.

We go on to show that the MAP optimally solves the Flexible Minmax Subset Problem (**FMSP**). In this generic combinatorial problem, a subset of weighted columns must be chosen, such that the aggregate weight exceeds *p*, and the sum of the row maxima is minimized. We prove that if we may permute each column independently, then an optimal solution can be found by ordering every column monotonically, and selecting among the ordered columns. We show that a single period of **CCNIM-MAP** can be transformed to an instance of **FMSP**. Algorithms for solving **FMSP** are already known in the literature, and include the MSP algorithm presented in Kress et al. [2004] and appropriately tailored enumeration schemes for "p-efficient points" (PEPS), e.g., Beraldi and Ruszczyński [2002].

We propose a practical algorithm, **CCNIM-dc,** that solves **CCNIM-MAP** for a wide range of cost ratios, assuming that the ships' missile capacities are identical. If the cost of storing missiles at the depot is only a little higher than the cost of allocating missiles to the ships, the resulting solution is not, generally, guaranteed optimal. However, **CCNIM-dc** solves much faster than the deterministic equivalent integer program, **RFFAM**, which often cannot solve problems of practical size for any cost ratio. **CCNIM-dc**, which we implement in Matlab version 7, solves most test-problem instances with as many as 24 ships and 24 scenarios in each period, in less than one minute of CPU-time on a Pentium IV, 2.8 GHz personal computer. If the result is not provably optimal, the optimality gap remaining is usually just a few percent.

A second advantage of **CCNIM-dc** is that, except when the cost ratio is 1, it provides all of the initial inventory combinations that solve the problem optimally. Planners can then select a preferred allocation according to other criteria not explicitly handled by the algorithm. Furthermore, **CCNIM-dc**, does not require specialized optimization software, and may be implemented in any generic computing language.

B. FUTURE WORK

A potentially important procurement policy would minimize the total number of missiles procured, and allocate as many of them as possible to the combat forces. This policy can be summarized by a cost ratio slightly greater than one. **CCNIM-dc** does not solve optimally for such a cost ratio, and such an extension may prove useful.

A key assumption of **CCNIM-MAP** is that the number of targets does not exceed the number of shooters in any scenario. To analyze the opposite situation with our techniques, the analyst must first predetermine which pairs of targets (or more) will comprise a single mission, so that the total number of missions dos not exceed the number of shooters. We are unaware of any optimal method for doing this except brute-force enumeration. Thus, extending **CCNIM-MAP** to handle this situation may prove to be an interesting challenge.

Flexible target assignments usually arise in naval combat, but in most situations, and in particular as force sizes grow, the assignments are not fully flexible nor are they necessarily one-to-one. Semi-flexible assignments pose a challenge to the current modeling scheme, particularly if legal target assignments are not known before combat actually occurs. Methods to approximate such cases should be developed, as solutions of **CCNIM-MAP** may be optimistic.

Another issue that merits attention arises from the fact that assigning targets in the real world is a stochastic process. In practice, not all of the targets in a scenario are detected at the time an attack begins, and some shooters will have to be committed to targets before the force commander can see the full extent of the attack. Consequently, targets may be identified or at least assessed incorrectly, causing the expenditure of too many or too few missiles. A special model or sub-model will be required to handle such situations (for example, see Washburn [2001]). A practical allocation plan should be robust to the effects of combat uncertainty, and techniques to ensure this should be explored.

Finally, we note the need for handling this issue: potential enemy interdiction of our assets. Interdiction may occur through (a) direct attack on the missile depots, or (b) disruption of access to those depots by submarines or offensive mining operations. Furthermore, missiles that are placed on ships may be lost in combat if those ships come under attack. In theory it is not too difficult to add scenarios to incorporate potential interdictions, but actually solving such a model might require substantial research effort. In any case, it seems that handling larger targets sets is a prerequisite to a consistent assessment of the effect of loss of shooters.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Ancker, C. J., (1982). *One-On-One Stochastic Duels*, Operations Research Society of America, Arlington, VA.

Appleget, J. A. and Wood, R. K., (2000). “Explicit-Constraint Branching for Solving Mixed-Integer Programs,” *Computing Tools for Modeling, Optimization and Simulation*, eds. Laguna, M. and Gonzalez-Velarde, J.L., Kluwer Academic Publishers, Boston, MA, pp. 243-261.

Avital, I., (2004). “Two-Period, Stochastic Supply-Chain Models with Recourse for Naval Surface Warfare,” MS in Operations Research, The Naval Postgraduate School, Monterey, CA.

Baker III, A. D., (2002). *The Naval Institute Guide to Combat Fleets of the World*, U.S. Naval Institute Press, Annapolis MD.

Bassok, Y., Hopp, W. J., and Rohatgi, M., (2002). “A Simple Linear Heuristic for the Service Constrained Random Yield Problem,” IIE Transactions, Vol. 34, No. 5, pp. 479-487.

Beraldi, P., and Ruszczyński, A., (2001). “A Branch and Bound Method for Stochastic Integer Problems under Probabilistic Constraints,” RUTCOR-Rutgers Center for Operations Research, RRR 16-2001, February.

Beraldi, P., and Ruszczyński, A., (2002). “The Probabilistic Set Covering Problem,” *Operations Research*, Vol 50, No 6, pp. 956-967.

Birge, J. and Louveaux, F., (1997). *Introduction to Stochastic Programming*, Springer-Verlag, New York, NY.

Charnes, C., Cooper, W. W., and Symonds, G. H. (1958). “Cost Horizons and Certainty Equivalents: an Approach to Stochastic Programming of Heating Oil Production,” *Management Science*, Vol. 4, No. 3, pp 235-263.

Conover, W. J. (1999). *Practical Nonparametric Statistics*, John Wiley and Sons, Inc., New York, NY.

Dentcheva, D., Prékopa, A., and Ruszczyński, A., (2000). "Concavity and Efficient Points of Discrete Distributions in Probabilistic Programming," *Math. Programming*, Ser. A 89, pp 55–77.

Diwekar, U. M., (2002). "Optimization under Uncertainty: an Overview," SIAG-OPT Views-and-News, 13, no. 1, pp. 1-8.

Hughes, W. P. Jr, (1995). "A Salvo Model of Warships in Missiles Combat Used to Evaluate Their Staying Power," *Warfare Modeling*, eds. Bracken, J., Kress, M. and Rosenthal, R. E., John Wiley and Sons, Inc., Danvers, MA, pp 121-144.

ILOG CPLEX. [http://www.ilog.com]. December 10, 2003.

Kress, M., (2002). *Operational Logistics – The Art and Science of Sustaining Military Operations*, Kluwer Academic Publishers, Boston, MA.

Kress, M., Penn, M., and Polukarov, M., (2004). "Two-Stage Supply-Chain with Recourse and Probabilistic Constraints," in review.

Miller, L.B., and Wagner, H., (1965). "Chance-Constrained Programming with Joint Constraints," *Operations Research* 13, pp. 930-945.

Murr, M. R. and Prékopa, A., (1996). "Solution of a Product Substitution Problem Using Stochastic Programming," RUTCOR-Rutgers Center for Operations Research, RRR 32-1996, November.

Paschalidis, I. C., Yong, L., Cassandras, C. G., and Panayiotou, C., (2004). "Inventory Control for Supply Chains with Service Level Constraints: A synergy between Large Deviations and Perturbation Analysis," *The Annals of Operations Research*, Vol. 126, pp. 231-258.

Porteus, E. L., (1990). "Stochastic Inventory Theory," *Handbooks in OR & MS, Vol. 2 - Stochastic Models*, eds. Heyman, D. P. and Sobel, M.J., Elsevier Science Publishers B. V., Amsterdam, pp. 605-652.

Prékopa, A., (1990). "Dual method for the Solution of One-Stage Stochastic Programming with Random Rhs Obeying a Discrete Probability Distribution," *Zeitschrift fur Operations Research* 34, pp. 441-461.

Prékopa, A., (1995). *Stochastic Programming*, Kluwer Academic Publishers, Boston, MA.

Prékopa, A., Vizvári, B. and Badics, T., (1996). "Programming Under Probabilistic Constraint With Discrete Random Variable," RUTCOR-Rutgers Center for Operations Research, RRR 10-1996, March.

Rabinovitch, A., (1997). *The Boats of Cherbourg*, Naval Institute Press, Annapolis, MD.

Rardin, R. L. (1998). *Optimization in Operations Research*, Prentice Hall, Upper Saddle River, NJ.

Ruszczyński, A., (2002). "Probabilistic Programming with Discrete Distributions and Precedence Constrained Knapsack Polyhedra," *Mathematical Programming* Ser. A 93, pp 195–215.

Simchi-Levi, D., Kaminsky, P., and Simchi-Levi, E., (2000). *Designing and Managing the Supply Chain*, Irwin McGraw-Hill, Boston, MA.

Washburn, A., (2001). *Joint Optimizing Informational Strike Tool*, NPS-OR-02-001-PR Project Report, Naval Postgraduate School, Monterey, CA.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A – FFAM

For comparison purposes, we provide the formulation of **FFAM**, the original IP used to solve **CCNIM**, which appeared in Avital [2004]. This formulation follows identical assumptions to those of **RFFAM**, but has many more binary variables. It uses auxiliary models, similar to those used by **RFFAM**, to generate some of the required parameters.

A. FFAM SPECIFICATION

The following non-mathematical description of FFAM itemizes a list of problem requirements followed by the constraint keys where they are represented in the mathematical model presented in the next section.

Regular constraints

- Minimize the weighted cost of allocating missiles to the combat ships in period I and storing extra missiles at a depot for possible use in period II, (7.1).

  Subject to:

- Each scenario in each period is successful only if every ship has enough missiles to satisfy the demand of its assigned mission, (7.2) and (7.17).
- In each period, the probability of successfully covering the scenarios exceeds a user-specified threshold, (7.3). In the second period, the cumulative probability must be achieved for every possible, preceding, period-I scenario, (7.18).
- Each ship is allocated a specific number of missiles in period I, (7.4).
- Each ship in each scenario is assigned one mission, (7.5) and (7.19).
- Each mission in each scenario is assigned to exactly one ship (with a specific number of missiles), (7.6) and (7.20).
- Following assignment and prosecution of a mission, each ship in each period-I scenario maintains an inventory equal to its initial level less the demand associated with its assigned mission, if that demand can be met, (7.7). Otherwise, the remaining inventory is zero, (7.8). (All ships “stay and fight.”)
- Each ship’s post-scenario inventory equals the number of missiles remaining in its inventory after prosecuting a mission, (7.9) and (7.10).
- For each ship, the number of missiles that may be placed on it is bounded from below (discretionary operational constraint) and from above (physical capacity limit), (7.11) and (7.21).

- For each period-I scenario, each ship's total inventory of missiles, before prosecuting any period-II mission, equals the post-mission inventory after period I plus any missiles that are replenished, (7.12)-(7.14).
- For each ship and each period-I scenario, only one level of replenishment may take place following period I, (7.15).
- Following each period-I scenario, the total number of missiles distributed to the ships between periods of combat may not exceed the number kept in the depot,(7.16) .

Specialized constraints

- In each scenario, missions with greater demands are assigned to ships with greater inventories (MAP constraints), (7.22) and (7.23).
- The number of missiles allocated to ship *i*+1 does not exceed the number allocated to ship *i*, (symmetry-breaking constraints) (7.24).
- The total number of missiles allocated in each period must exceed some minimum (total-allocation valid inequalities), (7.25)-(7.27).
- Each ship is allocated at least some minimum number of missiles in period I (single-ship valid inequalities), (7.28).

B. FFAM MATHEMATICAL DESCRIPTION

Indices

$i \in I$	ships
$k \in K$	level (number) of missiles ($K = \{0, \ldots, \max \overline{v}_i\}$)
$m \in M$	missions
$s \in \mathrm{S}^{\mathrm{I}}$	scenario s in period I
$s' \in \mathrm{S}^{\mathrm{II}}(s)$	scenario s' in period II
$\mathrm{S}^{\mathrm{II}}(s)$	Subset of period-II scenarios that may occur following period-I scenario *s*

Parameters [units]

φ_s	probability that period-I scenario *s* occurs
$\varphi_{s'\mid s}$	conditional probability that period-II scenario s' occurs, given period-I scenario *s* occurs
p^{I}	probability threshold for period I (probability that the realized scenario must be satisfied)

p_s^{II}	probability threshold for period II, if scenario *s* occurred in period I
d_{ms}	demand associated with mission *m* in scenario *s* [missiles]
c_1	unit cost of procuring a missile and allocating it to a ship [$/missile]
c_2	unit cost of procuring a missile and storing it in the depot [$/missile]
$\underline{v}_i$	discretionary lower bound on the number of missiles that may be allocated to ship *i* [missiles]
$\overline{v}_i$	physical upper bound on the number of missiles that may be allocated to ship *i* [missiles]

Auxiliary Parameters [units]

b^{I}	minimum aggregate ship load-out required to satisfy the period-I scenarios [missiles]
b_s^{II}	minimum aggregate ship load-out required to satisfy the period-II scenarios following scenario *s* in period I [missiles]
$\underset{\smile}{v}_i$	period-I data-specific lower bound on the number of missiles that may be allocated to ship *i* [missiles]

Decision Variables [units]

x_{ik}^{I}	1 if ship *i* has *k* missiles before the first combat period, and 0 otherwise
v_i	number of missiles initially allocated to ship *i* [missiles]
x^{II}	number of missiles allocated initially to the central depot [missiles]
r'_{ikms}	1 if ship *i* has *k* missiles remaining following mission *m* in period-I scenario *s*, and 0 otherwise
r_{iks}	1 if ship *i* has *k* missiles left following period-I scenario *s*, and 0 otherwise
w_{iks}	1 if ship *i* receives *k* missiles following period-I scenario *s*, and 0 otherwise
$x_{ikk's}$	1 if ship *i* has exactly k' missiles left after scenario *s*, and k'' missiles are added at the replenishment opportunity, and 0 otherwise
x_{iks}^{II}	1 if ship *i* is replenished to level *k* missiles following period-I scenario *s*, and 0 otherwise

z_s	1 if the demand vector in period-I scenario *s* is satisfied by the allocation plan, and 0 otherwise
$z_{s'\|s}$	1 if the demand vector in period-II scenario *s′* is satisfied by the allocation plan given that scenario *s* occurs in period I, and 0 otherwise
u^s_{ikm}	1 if ship *i* has *k* missiles and is assigned mission *m* in period-I scenario *s*, and 0 otherwise
$u^{s'\|s}_{ikm}$	1 if ship *i* has *k* missiles and is assigned mission *m* in period-II scenario *s′*, following period-I scenario *s*, and 0 otherwise

Formulation

$$\min_{\mathbf{u,x,y,z}} c_1 \sum_i \sum_k k\, x^{\mathrm{I}}_{ik} + c_2 x^{\mathrm{II}} \tag{7.1}$$

s.t.

Period I:

$$\sum_m \sum_{k \ge d_{ms}} u^s_{ikm} \ge z_s \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.2}$$

$$\sum_{s \in \mathbb{S}^{\mathrm{I}}} \varphi_s z_s \ge p^{\mathrm{I}} \tag{7.3}$$

$$\sum_k x^{\mathrm{I}}_{ik} = 1 \quad \forall i \tag{7.4}$$

$$\sum_m u^s_{ikm} = x^{\mathrm{I}}_{ik} \quad \forall i,\ k,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.5}$$

$$\sum_i \sum_k u^s_{ikm} = 1 \quad \forall m,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.6}$$

$$u^s_{i,k+d_{ms},m} = r'_{ikms} \quad \forall i,\ k \ge 1,\ m,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.7}$$

$$\sum_{k' \le d_{ms}} u^s_{i,k',m} = r'_{ikms} \quad \forall i,\ k = 0,\ m,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.8}$$

$$\sum_m r'_{ikms} = r_{iks} \quad \forall i,\ k,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.9}$$

$$\sum_k r_{iks} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.10}$$

$$x^{\mathrm{I}}_{ik} \equiv 0 \quad \forall i,\ k \mid \left(k < \underline{v}_i\right) \cup \left(k > \overline{v}_i\right) \tag{7.11}$$

All variables binary.

Period II:

$$x_{ikk's} \le r_{iks} \quad \forall i,\ k,\ k' \in K,\ s \in \mathbb{S}^{\mathrm{I}} \tag{7.12}$$

$$x_{ikk's} \le w_{ik's} \quad \forall i,\, k,\, k' \in K,\, s \in \mathbb{S}^{\mathrm{I}} \tag{7.13}$$

$$\sum_{k'} \sum_{k''|k'+k''=k} x_{ik'k''s} = x^{\mathrm{II}}_{iks} \quad \forall i,\; k,\; s \in \mathbb{S}^{\mathrm{I}} \tag{7.14}$$

$$\sum_{k} w_{iks} = 1 \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{7.15}$$

$$\sum_{i} \sum_{k} k\, w_{iks} \le x^{\mathrm{II}} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{7.16}$$

$$\sum_{m} \sum_{k \ge d_{ms'}} u^{s'|s}_{ikm} \ge z_{s'|s} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}},\, s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{7.17}$$

$$\sum_{s' \in \mathbb{S}^{\mathrm{II}}(s)} \varphi_{s'|s} z_{s'|s} \ge p^{\mathrm{II}}_{s} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{7.18}$$

$$\sum_{m} u^{s'|s}_{ikm} = x^{\mathrm{II}}_{iks} \quad \forall i,\, k,\, s \in \mathbb{S}^{\mathrm{I}},\, s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{7.19}$$

$$\sum_{i} \sum_{k} u^{s'|s}_{ikm} = 1 \quad \forall m,\, s \in \mathbb{S}^{\mathrm{I}},\, s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{7.20}$$

$$x^{\mathrm{II}}_{iks} \equiv 0 \quad \forall i,\, k \,|\, \left(k < \underline{v}_i\right) \cup \left(k > \overline{v}_i\right),\, s \in \mathbb{S}^{\mathrm{I}} \tag{7.21}$$

All variables binary except $x^{\mathrm{II}} \in \mathbb{Z}^{+}$.

Specialized constraints:

$$u^{s}_{ikm} = 0 \quad \forall i,\, k,\, m \ne i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{7.22}$$

$$\sum_{i} \sum_{k' \ge k} u^{s'|s}_{ik'm} \ge \sum_{i} \sum_{k' \ge k} u^{s'|s}_{ik',m+1} \quad \forall k,\, m \le n-1,\, s \in \mathbb{S}^{\mathrm{I}},\, s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{7.23}$$

$$\sum_{k' \ge k} x^{\mathrm{I}}_{ik'} \ge \sum_{k' \ge k} x^{\mathrm{I}}_{i+1,k'} \quad \forall i \le n-1,\, k \tag{7.24}$$

$$v_i = \sum_{k} k\, x^{\mathrm{I}}_{ik} \quad \forall i \tag{7.25}$$

$$\sum_{i} v_i \ge b^{\mathrm{I}}. \tag{7.26}$$

$$\sum_{i} \sum_{k} k\, x^{\mathrm{II}}_{iks} \ge b^{\mathrm{II}}_{s} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{7.27}$$

$$x^{\mathrm{I}}_{ik} \equiv 0 \quad \forall i,\, k < \underset{\smile}{v}_i \tag{7.28}$$

With these restrictions on the data:

$$\overline{v}_i \ge \overline{v}_{i+1} \;\; \forall i \le n-1 \tag{7.29}$$

$$d_{ms} \ge d_{m+1,s} \;\; \forall m \le n-1,\, \forall s \in \mathbb{S}^{\mathrm{I}} \cup \mathbb{S}^{\mathrm{II}}. \tag{7.30}$$

This model contains some variables that can be substituted out, namely x_{ik}^{I}, r_{iks}, x_{iks}^{II}, and v_i. Defining these variables in constraints (7.5), (7.9), (7.14), and (7.25) generates "branching constraints," however, and branching on these variables accelerates the branch-and-bound solution process for the integer model; see Appleget and Wood [2000].

APPENDIX B – SINGLE-PERIOD MIPS

Following are the formulations of the two auxiliary models used to set some of the parameters used in **RFFAM**. **RFFAM-sp** is a single-period model using the same assumptions as **RFFAM**. **RFFAM-lb** is a relaxation of **RFFAM-sp**. Both models use the same notation as **RFFAM**, and we provide it here for the reader's convenience. The equation numbers reference the first appearance of these constraints in Chapter II.

Indices

$i \in I$	ships
$k \in K$	level (number) of missiles
$m \in M$	missions
$s \in \mathrm{S}^{\mathrm{I}}$	scenario s in period I
$s' \in \mathrm{S}^{\mathrm{II}}(s)$	scenario s' in period II
$\mathrm{S}^{\mathrm{II}}(s)$	Subset of period-II scenarios that may occur following period-I scenario *s*

Parameters [units]

d_{ms}	demand associated with mission *m* in scenario *s* [missiles]
φ_s	probability that period-I scenario *s* occurs
$\varphi_{s'\|s}$	conditional probability that period-II scenario s' occurs, given that period-I scenario *s* occurs
p^{I}	probability threshold for period I (probability that the realized scenario must be satisfied)
p_s^{II}	probability threshold for period II, if scenario *s* occurs in period I
$\underline{v}_i$	discretionary lower bound on the number of missiles that may be allocated to ship *i* [missiles]
$\overline{v}_i$	physical upper bound on the number of missiles that may be allocated to ship *i* [missiles]

Decision Variables [units]

v_i^{I}	number of missiles allocated to ship *i* in period I [missiles]
x_{ik}^{I}	1 if ship *i* has *k* missiles before the first combat period, and 0 otherwise
v_{is}^{II}	number of missiles allocated to ship *i* in period II following period-I scenario *s* [missiles]
x_{iks}^{II}	1 if ship *i* is replenished to level *k* missiles following period-I scenario *s*, and 0 otherwise
z_s^{I}	1 if period-I scenario *s* is satisfied, and 0 otherwise
z_{is}^{I}	1 if ship *i* successfully covers its assigned mission in period-I scenario *s* , and 0 otherwise
$z_{s's}^{\mathrm{II}}$	1 if period-II scenario *s′* is successful following period-I scenario *s*, and 0 otherwise
$z_{is's}^{\mathrm{II}}$	1 if ship *i* successfully covers its assigned mission in period-II scenario *s′* following period-I scenario *s*, and 0 otherwise
u_{ims}^{I}	1 if ship *i* is assigned mission *m* in period-I scenario *s*, and 0 otherwise
$u_{ims's}^{\mathrm{II}}$	1 if ship *i* is assigned mission *m* in period-II scenario *s′* following period-I scenario *s*, and 0 otherwise

A. RFFAM-sp

$$\min_{\mathbf{u},\mathbf{v},\mathbf{x},\mathbf{z}} \sum_i v_i^{\mathrm{I}} \tag{2.16}$$

s.t.

$$v_i^{\mathrm{I}} = \sum_k k x_{ik}^{\mathrm{I}} \quad \forall i \tag{2.17}$$

$$z_{is}^{\mathrm{I}} \le \frac{1}{\overline{v}_i}\left(\sum_{k \ge d_{ms}} k x_{ik}^{\mathrm{I}} - d_{ms} u_{ims}^{\mathrm{I}}\right) + 1 \quad \forall i,\, m,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.18}$$

$$z_s^{\mathrm{I}} \le z_{is}^{\mathrm{I}} \quad \forall i,\, s \in \mathbb{S}^{\mathrm{I}} \tag{2.19}$$

$$\sum_{s \in \mathbb{S}^{\mathrm{I}}} \varphi_s z_s \ge p^{\mathrm{I}} \tag{2.20}$$

$$\sum_k x_{ik}^{\mathrm{I}} = 1 \quad \forall i \tag{2.21}$$

$$x_{ik}^{\mathrm{I}} \equiv 0 \quad \forall i,\, k \,|\, \left(k < \underline{v}_i\right) \cup \left(k > \overline{v}_i\right) \tag{2.29}$$

$$u_{iis}^{\mathrm{I}} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.41}$$

$$u_{im}^{s} = 0 \quad \forall i,\ m \neq i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.42}$$

$$\sum_{k' \geq k} x_{ik'}^{\mathrm{I}} \geq \sum_{k' \geq k} x_{i+1,k'}^{\mathrm{I}} \quad \forall i \leq n-1,\ k \tag{2.43}$$

All variables binary except v_i^{I}.

B. RFFAM-spII

$$\min_{\mathbf{u},\mathbf{x},\mathbf{v},\mathbf{z}} \sum_{s \in \mathbb{S}^{\mathrm{I}}} \sum_{i} v_{is}^{\mathrm{II}} . \tag{2.50}$$

s.t.

$$\sum_{k} k\, x_{iks}^{\mathrm{II}} = v_{is}^{\mathrm{II}} \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.31}$$

$$z_{is's}^{\mathrm{II}} \leq \frac{1}{\overline{v}_i} \left(\sum_{k \geq d_{ms'}} k\, x_{iks}^{\mathrm{II}} - d_{ms'} u_{ims's}^{\mathrm{II}} \right) + 1 \quad \forall i,\ m,\ s \in \mathbb{S}^{\mathrm{I}},\ s \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.32}$$

$$z_{s's}^{\mathrm{II}} \leq z_{is's}^{\mathrm{II}} \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.33}$$

$$\sum_{s' \in \mathbb{S}^{\mathrm{II}}(s)} \varphi_{s'|s} z_{s's}^{\mathrm{II}} \geq p_s^{\mathrm{II}} \quad \forall s \in \mathbb{S}^{\mathrm{I}} \tag{2.34}$$

$$\sum_{k} x_{iks}^{\mathrm{II}} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.35}$$

$$\sum_{m} u_{ims's}^{\mathrm{II}} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.36}$$

$$\sum_{i} u_{ims's}^{\mathrm{II}} = 1 \quad \forall m,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) \tag{2.37}$$

$$v_{is}^{\mathrm{II}} \in \left\{ \underline{v}_i, \underline{v}_i + 1, \ldots, \overline{v}_i \right\} \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.39}$$

$$x_{iks}^{\mathrm{II}} \equiv 0 \quad \forall i,\ k \mid \left(k < \underline{v}_i\right) \vee \left(k > \overline{v}_i\right),\ s \in \mathbb{S}^{\mathrm{I}} \tag{2.40}$$

$$u_{iis's}^{\mathrm{II}} = 1 \quad \forall i,\ s \in \mathbb{S}^{\mathrm{I}},\ s' \in \mathbb{S}^{\mathrm{II}}(s) . \tag{2.51}$$

All variables binary except v_{is}^{II}.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C – CASE SPECIFICATIONS

Following are tables listing the case specifics for the RFFAM cases run in Chapter II. In each case, $\varphi_s = 1/\left|\mathbb{S}^{\mathrm{I}}\right| \ \forall s$, $\varphi_{s'|s} = 1/\left|\mathbb{S}^{\mathrm{II}}\right| \ \forall s'$, $p^{\mathrm{I}} = 1 - 1/\left|\mathbb{S}^{\mathrm{I}}\right|$, and $p_s^{\mathrm{II}} = 1 - 1/\left|\mathbb{S}^{\mathrm{II}}\right| \ \forall s$. The capacity bounds are set at $\underline{v}_i = 2 \ \forall i$ and $\overline{v}_i = 8 \ \forall i$. For each case, the length of the demand vectors equals the number of ships, and we use the following indexing scheme: $s \in \mathbb{S}^{\mathrm{I}}$, $s' \in \mathbb{S}^{\mathrm{II}}$.

	s_1	s_2	s_3	s'_4	s'_5	s'_6
m_1	6	5	6	8	7	6
m_2	6	3	5	2	2	5
m_3	1	1	3	0	0	4

Table 7. Parameter Specifications for Case 2a.

	s_1	s_2	s_3	s_4	s'_5	s'_6	s'_7	s'_8
m_1	3	7	6	5	8	7	8	5
m_2	2	7	5	5	7	6	6	4
m_3	2	3	2	4	5	5	4	3
m_4	0	2	1	4	4	2	1	1

Table 8. Parameter Specifications for Case 2b.

	s_1	s_2	s_3	s_4	s_5	s'_6	s'_7	s'_8	s'_9	s'_{10}
m_1	6	5	8	6	8	7	8	5	7	7
m_2	5	5	6	5	4	7	7	4	2	5
m_3	3	5	5	4	4	4	3	2	1	4
m_4	1	3	3	3	2	2	2	2	0	2
m_5	0	3	1	1	2	1	2	2	0	1

Table 9. Parameter Specifications for Case 2c.

	s_1	s_2	s_3	s_4	s_5	s_6	s'_7	s'_8	s'_9	s'_{10}	s'_{11}	s'_{12}
m_1	5	7	7	8	7	8	5	8	8	8	8	7
m_2	5	7	7	7	5	7	4	7	7	8	5	6
m_3	4	6	7	6	5	7	2	5	4	8	5	6
m_4	4	4	6	6	5	6	2	1	1	7	3	4
m_5	3	4	3	5	3	6	0	1	1	6	1	3
m_6	0	3	0	0	0	5	0	1	0	5	0	1

Table 10. Parameter Specifications for Case 2d.

	s_1	s_2	s_3	s_4	s_5	s'_6	s'_7	s'_8	s'_9	s'_{10}
m_1	7	8	8	8	6	7	5	8	7	8
m_2	4	7	7	7	5	7	4	6	7	7
m_3	4	7	5	6	1	7	3	5	5	6
m_4	4	1	2	5	1	6	3	5	5	1
m_5	0	1	0	4	0	4	0	3	3	0

Table 11. Parameter Specifications for Case 2e.

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s'_9	s'_{10}	s'_{11}	s'_{12}	s'_{13}	s'_{14}	s'_{15}	s'_{16}
m_1	8	8	7	7	7	8	7	8	7	8	8	8	8	7	7	5
m_2	7	5	7	6	5	7	7	7	6	6	8	6	7	7	6	4
m_3	7	4	6	6	3	6	4	5	5	5	7	4	6	6	5	3
m_4	6	3	5	5	3	6	4	4	5	4	7	3	4	6	4	1
m_5	5	2	4	5	3	6	3	4	5	4	6	3	3	6	3	1
m_6	3	2	3	4	3	4	3	3	4	4	5	3	3	2	2	1
m_7	0	2	2	2	0	3	2	3	2	3	5	2	3	0	2	1
m_8	0	0	1	0	0	1	2	2	0	0	1	1	1	0	1	0

Table 12. Parameter Specifications for Case 2f.

APPENDIX D – CCNIM-e ALGORITHM

In this appendix we discuss the viability of a simple enumerative algorithm to solve **CCNIM-MAP** for cost ratios $c_2/c_1 \le 1$. Such an algorithm can overcome the no-safety-stock effect and find a point in $\mathtt{X}(\mathtt{F})$, as discussed in Chapter IV. We refer to this algorithm as **CCNIM-e**, where "e" stands for enumeration. Note that the validity of **CCNIM-e** is based on results derived in Chapter V.

We define an *atomic operation* as the procedure required to find an optimal allocation and depot inventory assuming that $p^{\mathrm{I}} = p^{\mathrm{II}} = 1$ and that $c_1 = c_2$. **CCNIM-e** consists of multiple repetitions of the atomic operation. Let S_q^{I} denote the q^{th} p^{I}-feasible subset of period-I scenarios, and let $\mathtt{Q}^{\mathrm{I}}$ denote the index set of all such subsets. For each scenario $s \in S^{\mathrm{I}}$, there is a (potentially different) set of period-II scenarios $S^{\mathrm{II}}(s)$ and probability requirement p_s^{II}. Let $\mathtt{Q}_s^{\mathrm{II}}$ denote the index set of the p^{II}-feasible subsets of $S^{\mathrm{II}}(s)$. The atomic operation is performed once for each combination of period-I subset S_q^{I} and period-II subsets corresponding to the scenarios $s \in S_q^{\mathrm{I}}$. The total number of operations N is given by

$$N = \sum_{q=1}^{|\mathtt{Q}^{\mathrm{I}}|} \prod_{s \in S_q^{\mathrm{I}}} \left|\mathtt{Q}_s^{\mathrm{II}}\right|, \tag{7.31}$$

which can obviously be quite large. The costs of all solutions are compared, using the actual cost coefficients, to find the optimal solution.

To assess the practicality of **CCNIM-e**, we need to obtain an estimate of the length of time required to solve the atomic operation, and calculate the number of times that operation must be repeated. The atomic operation is a special case of **CCNIM-MAP**, and can be solved by the following simple algorithm. It is described here in pseudo-code, given in vector notation, with vectors always being *n*-dimensional columns.

```
(1)  algorithm atomic
(2)  begin
(3)     $\mathbf{v}^{\mathrm{I}} := \max_s \{\mathbf{dd}_s^{\mathrm{I}}\}$;           // period-I allocation that satisfies every scenario
(4)     for $s := 1$ to $|S^{\mathrm{I}}|$ do
(5)        $\mathbf{r}_s = (\mathbf{v}^{\mathrm{I}} - \mathbf{d}_s^{\mathrm{I}})^+$;              // remaining inventories following scenario s
(6)        $\mathbf{r}_s := \mathrm{SORT}(\mathbf{r}_s, \text{'descend'})$;   // by inventory size (see Lemma 5)
(7)        $\mathbf{w}_s^{\mathrm{II}} = \max_{s' \in S^{\mathrm{II}}(s)} \{(\mathbf{dd}_{s'|s}^{\mathrm{II}} - \mathbf{r}_s)^+\}$;    // number of missiles to supplement each
     ship
(8)     end;
(9)     $x^{\mathrm{II}} :== \max_s \{\mathbf{1}^T \mathbf{w}_s^{\mathrm{II}}\}$;           // depot inventory
(10) end;
```

The complexity of the atomic operation can be analyzed as follows. The operation in line (3) requires one comparison for each element of each $\mathbf{dd}_s^{\mathrm{I}}$, and so is $\mathrm{O}(n|S^{\mathrm{I}}|)$. The operation in line (5) performs one subtraction and one comparison for each ship, and is, therefore, $\mathrm{O}(n)$. Sorting the vector in line (6) can be accomplished in $\mathrm{O}(n \log n)$ operations. The number of operations required in line (7) is $\mathrm{O}(n|S^{\mathrm{II}}(s)|)$. Because the loop of lines (4)-(8) is repeated $|S^{\mathrm{I}}|$ times, its complexity is given by $\mathrm{O}\left(n|S^{\mathrm{I}}|\left[\log n + \max_{s \in S^{\mathrm{I}}}\{|S^{\mathrm{II}}(s)|\}\right]\right)$, but in practical cases we expect the second term to dominate. Finally, in line (9) we also require $\mathrm{O}(n|S^{\mathrm{I}}|)$ operations. Therefore, we expect the computation time of the atomic operation to be governed, in practical cases, by $\mathrm{O}\left(n|S^{\mathrm{I}}|\max_{s \in S^{\mathrm{I}}}\{|S^{\mathrm{II}}(s)|\}\right)$. Note that we do not consider the computation of the vectors $\mathbf{dd}_s^{\mathrm{I}}$ and $\mathbf{dd}_{s'|s}^{\mathrm{II}}$ as part of the atomic operation, because these can be computed once for each instance of **CCNIM-e**.

We implement the atomic operation in MATLAB$^{\mathrm{TM}}$ version 6.5, and run it on an Intel 2.8 GHz Pentium IV personal computer. In order to obtain a fair comparison to the computation time required by **CCNIM-dc**, we draw and solve random instances of

CCNIM-MAP according to the same demand distribution we use in Chapter V. All demands for missiles are randomly generated, uniformly distributed integers on [0,8], and every scenario is equi-probable, with probabilities $\varphi^{\mathrm{I}} \equiv \varphi_s = 1/\left|S^{\mathrm{I}}\right| \;\; \forall s \in S^{\mathrm{I}}$ and $\varphi_s^{\mathrm{II}} \equiv \varphi_{s'|s} = 1/\left|S^{\mathrm{II}}(s)\right| \;\; \forall s' \in S^{\mathrm{II}}(s)$. We vary the value of n, $\left|S^{\mathrm{I}}\right|$, and $\left|S^{\mathrm{II}}(s)\right|$ between 3 and 30 in steps of size 3, for a total of 1000 observations. Because the timekeeping function in Matlab is inaccurate at measuring very short periods, we measure the time required to solve each instance 1000 times in succession. We derive the following regression function for the computation time of a single atomic operation, measured in micro-seconds.

$$t_{ao} = -23.13 + 1.45n + 18.95\left|S^{\mathrm{I}}\right| + 1.48\left|S^{\mathrm{II}}\right| + 0.1n\left|S^{\mathrm{I}}\right| - 0.08n\left|S^{\mathrm{II}}\right| + 0.06n\left|S^{\mathrm{I}}\right|\left|S^{\mathrm{II}}\right| \tag{7.32}$$

This model includes every interaction term except for $\left|S^{\mathrm{I}}\right|\left|S^{\mathrm{II}}\right|$, and appears to fit the data best, achieving $\mathrm{R}^2 = 0.9987$ and $F = 122{,}920$.

The estimate the computation of the enumerative algorithm, we simply need to multiply the computation time of the atomic operation of the correct size by the number of such operations required. When the scenarios are equi-probable and $\left|S^{\mathrm{II}}(s)\right| = \left|S^{\mathrm{II}}\right| \;\; \forall s \in S^{\mathrm{I}}$, the number of p-feasible subsets from a specific set is given by

$$|\mathcal{Q}| = \binom{|S|}{\lceil |S|\, p \rceil}. \tag{7.33}$$

Therefore, the number of times the atomic operation must be repeated to solve **CCNIM-MAP** is given by

$$N = \binom{\left|S^{\mathrm{I}}\right|}{\left\lceil \left|S^{\mathrm{I}}\right| p^{\mathrm{I}} \right\rceil} \binom{\left|S^{\mathrm{II}}\right|}{\left\lceil \left|S^{\mathrm{II}}\right| p^{\mathrm{II}} \right\rceil}^{\left\lceil \left|S^{\mathrm{I}}\right| p^{\mathrm{I}} \right\rceil}. \tag{7.34}$$

Although the atomic operation requires less than a millisecond to solve in problems of up to 20 ships and 20 scenarios in each period, it is evident that this procedure can only be used if the probability threshold is set such that every period-II scenario must be satisfied. Otherwise, the number of iterations is too large to solve in practical time. For example, in a case involving 10 scenarios in each period, and probability threshold of 0.9, we obtain $N = 10^{10}$ and $t_{ao} = 229.5\ \mu\sec$. The time required to solve that case using the enumeration technique is on the order of 26 days of computation.

Obviously we can do much better by identifying scenarios which are never candidates to be dropped, or scenarios which may be dropped every time. KPPs MSP algorithm, which we integrate into **CCNIM-dc**, uses these principles to avoid enumerating most of the period-II subsets. Furthermore, **CCNIM-dc** does not enumerate the p^{I}-feasible subsets at all; rather, it uses theoretical results based on PEPs to minimize the number of period-I candidate allocations we must explore.

APPENDIX E – CCNIM-dc ALGORITHM

Following is a pseudo-code representation of CCNIM-dc. When describing a particular function, the variables in square brackets represent the output of the function and the variables in parentheses following the function name represent the input to the function. All vector notation refers to *n*-dimensional vectors of ship properties, and all operations are component-wise. Functions are listed in order of appearance. Variables are local.

Algorithm $\left[cost,\ \hat{\mathtt{A}},\ \hat{\mathtt{X}}^{\mathrm{II}} \right]$ = **CCNIM-dc**

DATA

$i \in \{1,...,n\}$	ships
$S^{\mathrm{I}} = \left\{ s_1,...,s_{\lvert S^{\mathrm{I}} \rvert} \right\}$	period-I scenarios.
$S^{\mathrm{II}} = \left\{ s'_1,...,s'_{\lvert S^{\mathrm{II}} \rvert} \right\}$	period-II scenarios.
d_{is}^{I}	demand i in period I scenario *s*. $d_{is}^{\mathrm{I}} \geq d_{i+1,s}^{\mathrm{I}}\ \forall i \leq n-1$
$d_{is'}^{\mathrm{II}}$	demand i in period II scenario s'. $d_{is'}^{\mathrm{II}} \geq d_{i+1,s'}^{\mathrm{II}}\ \forall i \leq n-1$
p^{I}	probability threshold for period I.
p_s^{II}	probability threshold for period II, following period-I scenario *s*.
φ_s	probability of period-I scenario s.
$\varphi_{s'\mid s}$	conditional probability of period-II scenario s' on period-I scenario *s*.
$\overline{v}_i$	physical upper bound on missile capacity for ship *i*. $\overline{v}_i = \overline{v}\ \forall i$
$\underline{v}_i$	discretionary lower bound on missile capacity for ship *i*. $\underline{v}_i \geq \underline{v}_{i+1}\ \forall i \leq n-1$
c_1	cost of allocating a missile to a ship.
c_2	cost of allocating a missile to the depot.

OUTPUT

cost	cost of the proposed procurement plan
$\hat{\mathtt{A}}$	set of potential period-I allocations
$\hat{\mathtt{X}}^{\mathrm{II}}$	set of accompanying depot inventories

```
begin
  $\mathtt{D}^{\mathrm{I}} := \bigcup\{\mathbf{d}_s^{\mathrm{I}}\}$;                          // set of period-I scenarios
  $\mathtt{D}^{\mathrm{II}} := \bigcup\{\mathbf{d}_{s'}^{\mathrm{II}}\}$;                         // set of period-II scenarios
  for $s := 1$ to $|S^{\mathrm{I}}|$ do
    $\mathbf{dd}_s^{\mathrm{I}} := \max\{\mathbf{d}_s^{\mathrm{I}}, \underline{\mathbf{v}}\}$;                    // scenario requirements
  end;
  $\mathtt{DD}^{\mathrm{I}} := \bigcup \mathbf{dd}_s^{\mathrm{I}}$;
  $[\boldsymbol{\alpha}^{\mathrm{I}}, \boldsymbol{\beta}^{\mathrm{I}}] = \text{getPEPbounds}(\mathtt{DD}^{\mathrm{I}}, \varphi_s, p^{\mathrm{I}})$     // bounds on minimum allocation vector
  if $c_1 \geq c_2$ then
    $\mathtt{A} := \text{genAll}(\mathtt{DD}^{\mathrm{I}}, \varphi_s\ \forall s, p^{\mathrm{I}}, \boldsymbol{\alpha}^{\mathrm{I}}, \boldsymbol{\beta}^{\mathrm{I}})$    // generate period-I allocations
    $\mathtt{X}^{\mathrm{II}} := \text{depotWeight}(\mathtt{A}, \mathtt{D}^{\mathrm{I}}, \mathtt{D}^{\mathrm{II}}, \varphi^{\mathrm{II}}, p_s^{\mathrm{II}}, \underline{\mathbf{v}})$;  // and resulting depot inventory for each
    for $j := 1$ to $|\mathtt{A}|$ do
      $cost_j := c_1 \mathbf{1}^T \mathbf{v}_j^{\mathrm{I}} + c_2 \mathtt{X}_j^{\mathrm{II}}$;
    end;
    $\hat{j} := \text{argmin}\{\cos t_j\}$;                  // possibly multiple solutions
    $\hat{\mathtt{A}} := \mathbf{v}_{\hat{j}} \in \mathtt{A}$;        $\hat{\mathtt{X}}^{\mathrm{II}} := x_{\hat{j}}^{\mathrm{II}} \in \mathtt{X}^{\mathrm{II}}$;
  else
    $\mathtt{C} := \overline{\mathbf{v}}$;                              // initialize candidate allocations
    $\delta := n$;                               // initialize index of successor
    $x_d^{\mathrm{II}} := \text{depotWeight}(\mathtt{C}, \mathtt{D}^{\mathrm{I}}, \mathtt{D}^{\mathrm{II}}, \varphi^{\mathrm{II}}, p_s^{\mathrm{II}}, \underline{\mathbf{v}})$;     // calculate resulting depot inventory
    while $\mathtt{C} \neq \varnothing$
      $[\mathtt{CN}, \delta\mathtt{N}]$ :=prevLevel$(\mathtt{C}, \delta, \boldsymbol{\beta}^{\mathrm{I}}, \mathtt{DD}^{\mathrm{I}}, \varphi_s, p^{\mathrm{I}})$;
      $\mathtt{X}^{\mathrm{II}} := \text{depotWeight}(\mathtt{CN}, \mathtt{D}^{\mathrm{I}}, \mathtt{D}^{\mathrm{II}}, \varphi^{\mathrm{II}}, p_s^{\mathrm{II}})$;
      for $j := |\mathtt{CN}|$ to 1 by -1 do
        if $\mathtt{X}_j^{\mathrm{II}} > x_d^{\mathrm{II}}$ then
          $\mathtt{CN} := \mathtt{CN} \backslash \{\mathbf{u}_j\}$;                  // delete allocations that increase depot
        end;
      end;
      if $\mathtt{CN} == \varnothing$ then
        $\hat{\mathtt{A}} := \mathtt{C}$;                          // record minimum period-I allocations
      end;
      $\mathtt{C} := \mathtt{CN}$;
    end;
    $cost := c_1 \mathbf{1}^T \hat{\mathtt{A}}_1 + c_2 x_d^{\mathrm{II}}$;    $\hat{\mathtt{X}}^{\mathrm{II}} = x_d^{\mathrm{II}}$;
  end;
end;
```

function $[\boldsymbol{\alpha}, \boldsymbol{\beta}] = \text{getPEPbounds}(\mathtt{D}, \varphi_s\ \forall s,\ p,\ n)$

DESCRIPTION:

The function calculates the lower and upper bounds on the PEPs of the distribution of $\mathtt{D}$. The lower bounds are calculated through the marginal distribution, and the upper bounds from the largest value possible.

INPUT:

$\mathtt{D}$	set of scenarios
φ_s	scenario probability
p	probability threshold
n	number of ships

OUTPUT:

$\boldsymbol{\alpha}$	vector of PEP lower bounds
$\boldsymbol{\beta}$	vector of PEP upper bounds

```
begin
  for $i := 1$ to $n$ do
    let $\pi^i$ be a permutation of the scenarios such that $d^{\mathrm{I}}_{i,\pi^i(1)} \geq \ldots \geq d^{\mathrm{I}}_{i,\pi^i(|\mathtt{D}|)}$;
    $t^i := \operatorname{argmax}_t \sum_{\pi^i(s)=t}^{n} \varphi_{\pi^i(s)} \geq p$;      //  permuted index of threshold-passing  scenario
    $\alpha_i := d_{it^i}$;                          //  PEP lower bound based on marginal distribution
    $\beta_i := \max_s \{d_{is}\}$;                  //  PEP upper bound – highest demand possible
  end;
end;
```

function $[\mathtt{A}] = \text{genAll}\left(\mathtt{DD}^{\mathrm{I}},\ \varphi_s\ \forall s,\ p^{\mathrm{I}},\ \boldsymbol{\alpha}^{\mathrm{I}},\ \boldsymbol{\beta}^{\mathrm{I}}\right)$

DESCRIPTION:

This function generates all the period I allocations that need to initially be considered. By theorem 5.1, if $c_1 \geq c_2$, the candidate allocations are the PEPs of the distribution of $\mathbf{dd}_s^{\mathrm{I}}$, the allocation vector required for period I. Otherwise, we load the ships to capacity so we can calculate the minimum depot.

INPUT:

$\mathtt{DD}^{\mathrm{I}}$	period-I scenario requirements
φ_s	period-I scenario probabilities
p^{I}	period-I probability threshold
$\boldsymbol{\alpha}^{\mathrm{I}}$	ship inventory lower bounds
$\boldsymbol{\beta}^{\mathrm{I}}$	ship capacity upper bounds

OUTPUT:

$\mathtt{A}$	set of period-I allocation vectors

```
begin;
  $\mathtt{C} := [\boldsymbol{\alpha}^{\mathrm{I}}]$;                      // initialize set of candidate PEPs
  $\mathtt{A} := \varnothing$;                         // initialize set of known PEPs
  $\delta := 1$;                          // initialize set of predecessor indices
  while $\mathtt{C} \neq \varnothing$
    for $j := 1$ to $|\mathtt{C}|$ do             // check p-feasibility of each candidate
      $\mathbf{u} = \mathtt{C}_j$;
      $mk := \text{checkPF}(\mathbf{u},\ \mathtt{DD}^{\mathrm{I}},\ \varphi_s\ \forall s,\ p^{\mathrm{I}})$;
      if $mk == 1$ then
        $\mathtt{A} := \mathtt{A} \cup \{\mathbf{u}\}$;          // include in set of PEPs
        $\mathtt{C} := \mathtt{C} \setminus \{\mathbf{u}\}$;           // remove from set of candidates
      end;
    end;
    if $\mathtt{C} \neq \varnothing$ then
      $[\mathtt{C},\ \delta] := \text{NextGen\_PI}(\mathtt{C},\ \delta,\ \mathtt{A},\ \boldsymbol{\beta}^{\mathrm{I}},\ n)$;     // generate next set of candidates
    end;
  end;
end;
```

function [*bool*]:=checkPF$\left(\mathbf{v}, \mathbb{D}, \varphi_s\ \forall s,\ p\right)$

DESCRIPTION:

The function calculates the cumulative probability of scenarios which are successfully covered by $\mathbf{v}$, and checks whether the probability threshold is exceeded.

INPUT:

$\mathbf{v}$	candidate allocation vector
$\mathbb{D}$	set of scenarios
φ_s	scenario probability
p	probability threshold

OUTPUT:

bool is set to 1 if $\mathbf{v}$ is *p*-feasible

```
begin
   bool:=0;
   for $s := 1$ to $|\mathbb{D}|$ do
      $z_s := 0$;
      if $\mathbf{v} \geq \mathbf{d}_s$ then
         $z_s := 1$;
      end;
   end;
   if $\sum z_s \varphi_s \geq p$ then
      bool:=1;
   end;
end;
```

function $[\mathtt{CN}, \mathtt{\delta N}]$:=NextGen_PI$(\mathtt{C}, \mathtt{\delta}, \mathtt{A}, \boldsymbol{\beta}, n)$

DESCRIPTION:

This function generates the next generation of candidate PEPs. Lower bounds are based on the marginal distribution of each ship's demands. This enumeration scheme follows that described by Beraldi and Ruszczyński [2002], with additional conditions that guarantee the candidates are monotonic and the upper bound is maintained.

INPUT:

$\mathtt{C}$	set of failed candidates
$\mathtt{\delta}$	their predecessor indices
$\mathtt{A}$	current known PEPs
$\boldsymbol{\beta}$	PEP upper bounds
n	number of ships

OUTPUT:

$\mathtt{CN}$	next set of candidate PEPs
$\mathtt{\delta N}$	their predecessor indices

```
begin;
  $\mathtt{CN} := \varnothing$;                          // initialize set of candidates for next generation
  $\mathtt{\delta N} := \varnothing$;                          // initialize set of predecessor indicators
  for $j := 1$ to $|\mathtt{C}|$ do
    for $j' := \delta_j$ to $n$ do
      $\mathbf{u} := \mathtt{C}_j$;
      $u_{j'} := u_{j'} + 1$;                 // increase one component only
      if $(u_{j'} \le u_i \ \forall i \le j'-1)$ and $(u_{j'} \le \beta_{j'})$ then
        $\mathtt{CN} := \mathtt{CN} \cup \{\mathbf{u}\}$;          // if conditions are met, add to list of next generation
        $\mathtt{\delta N} := \mathtt{\delta N} \cup \{j'\}$;          // and store its predecessor index
      end;
    end;
  end;
  for $j := 1$ to $|\mathtt{CN}|$ do
    $\mathbf{u} := \mathtt{CN}_j$;
    $mk_j := 0$;                     // initialize domination indicator
    $j' := 1$;
    while $j' \le |\mathtt{A}|$
      $\mathbf{v} = \mathtt{A}_{j'}$;
      if $\mathbf{u} \ge \mathbf{v}$ then             // compare each candidate to list of known PEPs
        $mk_j := 1$;               // record those which are dominated by any PEP
        $j' := |\mathtt{A}| + 1$;            // exit internal loop
      else
```

```
                $j' := j' + 1$;          // index next known PEP
            **end**;
        **end**;
    **end**;
    **if** $mk == 1$ **then**
        $\mathtt{CN} := \mathtt{CN} \setminus \{\mathbf{u}\}$;      //  delete the dominated candidate from the list
        $\delta\mathtt{N} := \delta\mathtt{N} \setminus \{ j' \}$;     //  and its predecessor index
    **end**;
**end**;
```

function $\left[\mathtt{X}^{\mathrm{II}}\right]$ = depotWeight$\left(\mathtt{A}, \mathtt{D}^{\mathrm{I}}, \mathtt{D}^{\mathrm{II}}, \varphi^{\mathrm{II}}_{s'|s}\ \forall s, s',\ p^{\mathrm{II}}_s, \underline{\mathbf{v}}\right)$

DESCRIPTION:

This function calculates the remaining inventories on each ship, reorders the remainders, and calculates the number of additional missiles each ship needs to successfully cover each period-II scenario. It then calls minPEP() to calculate the minimum period-II p-efficient point on the distribution of $\mathbf{dd}_{s'|s,j}$. minPEP() is a generic name, and actually, we call either the function MSPA() or minPEPenum() to obtain the minimum PEP.

INPUT:

$\mathtt{A}$	set of possible period-I allocations
$\mathtt{D}^{\mathrm{I}}$	set of period-I scenarios
$\mathtt{D}^{\mathrm{II}}$	set of period-II scenarios
$\varphi^{\mathrm{II}}_{s'\|s}$	conditional probabilities of period-II scenarios
p^{II}_s	probability threshold for period II, following period-I scenario *s*
$\underline{\mathbf{v}}$	ship's minimum inventories

OUTPUT:

$\mathtt{X}^{\mathrm{II}}$	set of depot inventories calculated for each period-I allocation

```
begin
   for $j := 1$ to $|\mathrm{A}|$ do
      for $s := 1$ to $|\mathtt{D}^{\mathrm{I}}|$ do
         $\mathbf{r}_{sj} = \left(\mathbf{v}^{\mathrm{I}}_j - \mathbf{d}^{\mathrm{I}}_s\right)^+$;                // remaining inventory
         $\mathbf{r}_{sj} := \mathrm{SORT}_i\left(\mathbf{r}_{sj}, \text{'descend'}\right)$;        // see theorem 5.2
         for $s' := 1$ to $|\mathtt{D}^{\mathrm{II}}|$ do
            $\mathbf{dd}^{\mathrm{II}}_{s'|s,j} = \left(\max\left\{\mathbf{d}^{\mathrm{II}}_{s'}, \underline{\mathbf{v}}\right\} - \mathbf{r}_{sj}\right)^+$;   // period-II requirements from the depot
         end;
         $\mathtt{DD}^{\mathrm{II}}_{sj} := \bigcup_{s'|\varphi^{\mathrm{II}}_{s'|s}>0} \left\{\mathbf{dd}^{\mathrm{I}}_{s'|s,j}\right\}$;          // set of period-II requirements
         $w^{\mathrm{II}}_{sj} := \mathrm{minPEP}\left(\mathtt{DD}^{\mathrm{II}}_{sj}, \varphi^{\mathrm{II}}_{s'|s} > 0,\ p^{\mathrm{II}}_s\right)$;   // optimal requirements given period-I
                                                          //   scenario s and allocation j
      end;
      $x^{\mathrm{II}}_j := \max_s\left\{w^{\mathrm{II}}_{sj}\right\}$;                  // the depot required for allocation j
   end;
   $\mathtt{X}^{\mathrm{II}} = \bigcup\left\{x^{\mathrm{II}}_j\right\}$;                     // set of depot inventories
end;
```

function $[k] = \text{minPEPenum}(\text{D}, \varphi_s\ \forall s,\ p)$

DESCRIPTION:

This function generates the value of the minimum level PEP for period II requirements through the PEP enumeration technique described in Beraldi and Ruszczyński [2002].

INPUT:

D	period-I scenario demands
φ_s	period-I scenario probabilities
p	period-I probability threshold

OUTPUT:

k	sum of elements of the minimum PEP

```
begin;
  $[\boldsymbol{\alpha}, \boldsymbol{\beta}] = \text{getPEPbounds}(\text{D}, \varphi_s, p)$
  $k := \mathbf{1}^T \boldsymbol{\alpha}$;                    // initial level
  $\text{C} := [\boldsymbol{\alpha}]$;                     // initialize set of candidate PEPs
  $\text{A} := \varnothing$;                       // initialize set of known PEPs
  $\delta := 1$;                       // initialize set of predecessor indices
  while $\text{A} = \varnothing$
    for $j := 1$ to $|\text{C}|$ do             // check p-feasibility of each candidate
      $\mathbf{u} = \text{C}_j$;
      $mk := \text{checkPF}(\mathbf{u}, \text{D}, \varphi_s\ \forall s,\ p)$;
      if $mk == 1$ then
        $\text{A} := \text{A} \cup \{\mathbf{u}\}$;             // include in set of PEPs
      end;
    end;
    if $\text{A} = \varnothing$ then
      $[\text{C}, \delta] := \text{NextGen\_PII}(\text{C}, \delta, \text{A}, \boldsymbol{\beta})$;      // generate next set of candidates
    end;
  end;
end;
```

function $[k] = \text{MSPA}(\mathtt{D}, \varphi_s \ \forall s, \ p)$

DESCRIPTION:

This function generates the value of the minimum level PEP for period II requirements by using the MSP algorithm developed by Kress et al. [2004]. The minimal p-feasible subset is created in a process of elimination. In the preliminary phase, scenarios that require too many missiles in total to be included in the minimal p-feasible subset are eliminated. In the second stage, p-feasible subsets are enumerated, and the one requiring a minimum number of missiles is selected.

INPUT:

$\mathtt{D}$	period-I scenario demands
φ_s	period-I scenario probabilities
p	period-I probability threshold

OUTPUT:

k	sum of elements of the minimum PEP

```
begin;
  for $s := 1$ to $|\mathtt{D}|$ do
    $\sigma_s := \mathbf{1}^T \mathbf{d}_s$;
  end;
  let $\pi_\sigma$ be a permutation of the scenarios such that $\sigma_{\pi_\sigma(1)} \geq \ldots \geq \sigma_{\pi_\sigma(|\mathtt{D}|)}$;
  $t_\sigma := \underset{t}{\text{argmax}} \sum_{\pi_\sigma(s)=t}^{n} \varphi_{\pi_\sigma(s)} \geq p$;      // permuted index of threshold-passing scenario
  for $i := 1$ to $n$ do
    $\alpha_i := d_{it_\sigma}$;                 // PEP lower bound based on marginal distribution
    $S^\sigma := \{ s \mid \pi_\sigma(s) \geq t_\sigma \}$;
    $\beta_i := \max_{s \in S^\sigma} \{ d_{is} \}$;           // nominal allocation
  end;
  $k := \mathbf{1}^T \boldsymbol{\beta}$;                   // number of missiles in nominal allocation
  $\mathtt{B} := \bigcup_{s \in S^\sigma} \{ \mathbf{d}_s \}$;            // potentially optimal p-feasible scenarios
  $p_{\mathtt{D} \setminus \mathtt{B}} := \sum_{s \notin S^\sigma} \varphi_s$;           // probability of eliminated scenarios
  $\Psi := \varnothing$;                    // initialize suspect scenarios
  for $i := 1$ to $n$ do
    let $\pi^i$ be a permutation of the scenarios $s \in S^\sigma$ such that $d^{\mathrm{I}}_{i,\pi^i(1)} \geq \ldots \geq d^{\mathrm{I}}_{i,\pi^i(|\mathtt{B}|)}$;
    $t^i := \underset{t}{\text{argmax}} \sum_{\pi^i(s)=1}^{t} \varphi_{\pi^i(s)} \leq 1 - (p + p_{\mathtt{D} \setminus \mathtt{B}})$;
    if $t^i \neq \varnothing$ then
      $\Psi := \Psi \cup \{ s \mid \pi^i(s) \leq t^i \}$;
```

```
        end;
    end;
    if $\Psi \neq \varnothing$ then
        $\kappa := \min\left\{\left\lceil 1-\left(p+p_{D\setminus B}\right) \big/ \min_{s\in\Psi}\varphi_s \right\rceil, |\Psi|\right\}$;   // number of scenarios we may eliminate
        for $j := 1$ to $\kappa$ do
            let $\Phi^j$ be the set of all subsets size j of scenarios from $\Psi$   ;
        end;
        $\Phi := \bigcup\left\{\Phi^j\right\}$;                                  //  set of all possibly eliminated subsets
        $cm := \sum_{j=1}^{\kappa}\binom{|\Psi|}{j}$;                                  //  cardinality of $\Phi$
        for $j := 1$ to $cm$ do
            if $\sum_{s\in\Phi_j}\varphi_s \leq 1-\left(p+p_{D\setminus B}\right)$ then            //  if the subset can be eliminated
                $k' := \sum_i \max_{s\in B\setminus\Phi_j}\left\{d_{is}\right\}$;              //  calculate allocation cost of remaining scenarios
                if $k' < k$ then
                    $k := k'$;                                // replace if improving
                end;
            end;
        end;
    end;
end;
```

function $[\mathtt{CN}, \mathtt{\delta N}]$:=NextGen_PII$(\mathtt{C}, \mathtt{\delta}, \boldsymbol{\beta}, n)$

DESCRIPTION:

This function generates the next generation of candidate PEPs in period II. It differs from the period-I generation in the fact that there is no monotonicity condition. We also need not eliminate dominated candidates, because this function is only used until the first PEP is found.

INPUT:

$\mathtt{C}$	set of failed candidates
$\mathtt{\delta}$	their predecessor indices
$\boldsymbol{\beta}$	PEP upper bounds
n	number of ships

OUTPUT:

$\mathtt{CN}$	next set of candidate PEPs
$\mathtt{\delta N}$	their predecessor indices

```
begin;
  $\mathtt{CN} := \varnothing$;                          // initialize set of candidates for next generation
  $\mathtt{\delta N} := \varnothing$;                          // initialize set of predecessor indicators
  for $j := 1$ to $|\mathtt{C}|$ do
    for $j' := \delta_j$ to $n$ do
      $\mathbf{u} := \mathtt{C}_j$;
      $u_{j'} := u_{j'} + 1$;                  // increase one component only
      if $(u_{j'} \le \beta_{j'})$ then
        $\mathtt{CN} := \mathtt{CN} \cup \{\mathbf{u}\}$;          // if conditions are met, add to list of next generation
        $\mathtt{\delta N} := \mathtt{\delta N} \cup \{j'\}$;          // and store its predecessor index
      end;
    end;
  end;
  if $mk == 1$ then
    $\mathtt{CN} := \mathtt{CN} \setminus \{\mathbf{u}\}$;          // delete the dominated candidate from the list
    $\mathtt{\delta N} := \mathtt{\delta N} \setminus \{j'\}$;          // and its predecessor index
  end;
end;
```

function $[\mathtt{CN}, \delta\mathtt{N}] := \text{prevLevel}\left(\mathtt{C}, \delta, \boldsymbol{\beta}^{\mathrm{I}}, \mathtt{DD}^{\mathrm{I}}, \varphi_s\ \forall s,\ p^{\mathrm{I}}, n\right)$

DESCRIPTION:

This function generates previous level of candidate period-I allocations. This enumeration scheme follows the reverse PEP enumeration described by Beraldi and Ruszczyński [2002]. Fewer points are enumerated because of the monotonicity requirement. The PEP upper bounds are used as a quick method to verify p-feasibility.

INPUT:

$\mathtt{C}$	set of current known allocations
δ	their predecessor indices
$\boldsymbol{\beta}^{\mathrm{I}}$	PEP upper bounds
$\mathtt{DD}^{\mathrm{I}}$	Set of period-I required ship inventories
φ_s	period-I scenario probabilities
p^{I}	period-I probability threshold
n	number of ships

OUTPUT:

$\mathtt{CN}$	next set of candidate allocations
$\delta\mathtt{N}$	their predecessor indices

```
begin;
  $\mathtt{CN} := \varnothing$;                    //  initialize set of candidates for next generation
  $\delta\mathtt{N} := \varnothing$;                    //  initialize set of predecessor indicators
  for $j := 1$ to $|\mathtt{C}|$ do
    for $j' := \max\{\delta_j - 1, 1\}$ to $\delta_j$ do
      $\mathbf{u} := \mathtt{C}_j$;
      $u_{j'} := u_{j'} - 1$;                //  reduce one component only
      if $(u_{j'-1} \geq u_{j'} \geq u_{j'+1})$ then   //  verify monotonicity of allocation is maintained
        $mk := 0$;
        if $\mathbf{u} \geq \boldsymbol{\beta}^{\mathrm{I}}$ then
          $mk := 1$;
        else
          $mk := \text{checkPF}(\mathbf{u}, \mathtt{DD}^{\mathrm{I}}, \varphi_s\ \forall s,\ p^{\mathrm{I}})$;
        end;
        if $mk == 1$ then
          $\mathtt{CN} := \mathtt{CN} \cup \{\mathbf{u}\}$;       //  if conditions are met, add to list of candidates
          $\delta\mathtt{N} := \delta\mathtt{N} \cup \{j'\}$;         //  and store its predecessor index
        end;
      end;
    end;
  end;
end;
```

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor R. Kevin Wood, Code OR/Wd
   Department of Operations research
   Naval Postgraduate School
   Monterey, California

4. Professor Moshe Kress, Code OR
   Department of Operations Research
   Naval Postgraduate School
   Monterey, California

5. Professor Andrzej Ruszczyński,
   Department of Management Science
   Rutgers University
   Piscataway, New Jersey

6. Ittai Avital
   Israel Navy
   Tel Aviv, Israel